CLAUDE BOLDUC

# INTERPROCEDURAL PROGRAM ANALYSIS USING VISIBLY PUSHDOWN KLEENE ALGEBRA

Thèse présentée
à la Faculté des études supérieures de l'Université Laval
dans le cadre du programme de doctorat en informatique
pour l'obtention du grade de Philosophiæ Doctor (Ph.D.)

FACULTÉ DES SCIENCES ET DE GÉNIE
UNIVERSITÉ LAVAL
QUÉBEC

2011

# Résumé

Les analyses interprocédurales automatiques de programmes qui sont basées sur des théories mathématiques rigoureuses sont complexes à réaliser, mais elles sont d'excellents outils pour augmenter notre confiance envers les comportements possibles d'un programme. Les méthodes classiques pour réaliser ces analyses sont l'analyse de modèles, l'interprétation abstraite et la démonstration automatique de théorèmes. La base d'un démonstrateur automatique de théorèmes est une logique ou une algèbre et le choix de celle-ci a un impact sur la complexité de trouver une preuve pour un théorème donné.

Cette dissertation développe un formalisme algébrique concis pouvant être utilisé en démonstration automatique de théorèmes. Ce formalisme est appellé *algèbre de Kleene à pile visible*. Cette dissertation explique comment ce formalisme peut être utilisé pour réaliser des analyses interprocédurales de programmes, comme des vérifications formelles et des vérifications d'optimisations effectuées par des compilateurs. Cette dissertation apporte aussi des preuves que ces analyses pourraient être automatisées.

L'algèbre de Kleene à pile visible est une extension de l'algèbre de Kleene, un excellent formalisme pour réaliser des analyses intraprocédurales de programmes. En bref, l'algèbre de Kleene est la théorie algébrique des automates finis et des expressions régulières. Donc, cette algèbre à elle seule n'est pas appropriée pour faire des analyses interprocédurales de programmes car la puissance des langages non contextuels est souvent nécessaire pour représenter le flot de contrôle d'un tel programme. L'algèbre de Kleene à pile visible étend celle-ci en lui ajoutant une famille d'opérateurs de plus petit point fixe qui est basée sur une restriction des grammaires non contextuelles. En fait, cette algèbre axiomatise exactement la théorie équationnelle des langages à pile visibles. Ces langages sont une sous-classe des langages non contextuels et ont été définis par Alur et Madhusudan pour faire de l'analyse de modèles. La complexité résultante de la théorie équationnelle de l'algèbre proposée est EXPTIME-complète.

# Abstract

Automatic interprocedural program analyses based on rigorous mathematical theories are complex to do, but they are great tools to increase our confidence in the behaviour of a program. Classical ways of doing them is either by model checking, by abstract interpretation or by automated theorem proving. The basis of an automated theorem prover is a logic or an algebra and the choice of this basis will have an impact in the complexity of finding a proof for a given theorem.

This dissertation develops a lightweight algebraic formalism for the automated theorem proving approach. This formalism is called visibly pushdown Kleene algebra. This dissertation explains how to do some interprocedural program analyses, like formal verification and verification of compiler optimizations, with this formalism. Evidence is provided that the analyses can be automated.

The proposed algebraic formalism is an extension of Kleene algebra, a formalism for doing intraprocedural program analyses. In a nutshell, Kleene algebra is the algebraic theory of finite automata and regular expressions. So, Kleene algebra alone is not well suited to do interprocedural program analyses, where the power of context-free languages is often needed to represent the control flow of a program. Visibly pushdown Kleene algebra extends Kleene algebra by adding a family of implicit least fixed point operators based on a restriction of context-free grammars. In fact, visibly pushdown Kleene algebra axiomatises exactly the equational theory of visibly pushdown languages. Visibly pushdown languages are a subclass of context-free languages defined by Alur and Madhusudan in the model checking framework to model check interprocedural programs while remaining decidable. The resulting complexity of the equational theory of visibly pushdown Kleene algebra is EXPTIME-complete whereas that of Kleene algebra is PSPACE-complete.

# Acknowledgements

During my graduate studies, I had the pleasure to be in excellent company. First of all, I am grateful to my advisor Béchir Ktari for his first-class guidance throughout my thesis. I will always remember his passion for research, his original thoughts on so many subjects and our animated discussions even late in the evening. More than that, I am very grateful that he left me enough autonomy to try out my (sometimes crazy) ideas.

My co-advisor Jules Desharnais is a pillar of the success of my graduate studies. Thank you for believing in me since the beginning of my college years! You showed me (I hope I understood!) how to become a better researcher by small details, hard work, creativity and by helping others.

I am grateful to my thesis committee and to the anonymous referees of articles written throughout my graduate studies. They helped to enhance this dissertation through their thoughtful comments. Also, I greatly appreciated the financial support of NSERC (Natural Sciences and Engineering Research Council of Canada) and FQRNT (Fond québécois de la recherche sur la nature et les technologies).

I spent a lot of time discussing with several (past and current) members of the LSFM research group. They provided inspiration, hints and funny jokes! A special thanks to François who was always willing to talk about algebraic structures and life, and to Amélie who was always willing to talk about life and algebraic structures.

Thanks to Félix-Antoine Bourbonnais for his friendship and to be never afraid to raise practical questions in my work.

My college years were a roller coaster of emotions! Thanks to my friends who decreased the downslopes and increased the upslopes, and to my parents who helped to keep balance by providing a fixed point. My biggest thanks to my exquisite wife, Véronique, who shared the whole ride laughing and screaming with me! She also provided invaluable feedback throughout my studies.

*To Véronique,*
*who makes each day more beautiful.*
*&*
*To Olivier,*
*my shining star in the sky,*
*who looks upon his dad.*
*You showed me determination,*
*courage and will to live.*
*I miss you!*
*&*
*To Cédrick,*
*the upcoming one,*
*may you be blessed*
*with all things good...*

*It is a curious fact that the algebras*
*that have been most extensively studied*
*in conventional (albeit modern!) algebra*
*do not have fundamental operations*
*of arity greater than two.*
*— Burris, Sankappanavar*

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

Separation of concerns is one of the most important design principle in object-oriented software engineering. One typical way to achieve this is through modularity of programming. Hence, functions (or code blocks or procedures or methods) are unavoidable for writing clean code for complex systems. Programs using functions are called *interprocedural programs* and programs not using them are called *intraprocedural programs*.

Analyzing an imperative interprocedural program for bugs is a difficult task, but it must be done to increase our confidence in the behaviour of a program. Currently, software testing and code review are the usual ways to analyze programs in the industry. Software testing proved itself useful, particularly in Agile software development, and even inspired software development techniques like test-driven development [4] and behaviour-driven development [42]. I believe that software testing is a necessary first step. However, software engineers should also seek to *complement* (not replace) software testing with other analyses to improve the quality of their software. Automatic interprocedural program analyses based on rigorous mathematical theories are a promising avenue. Among these, the following two are useful and complementary:

**Formal verification of interprocedural programs:** Proving the correctness of an interprocedural program with respect to a property (specification).

**Verification of interprocedural compiler optimizations:** Proving that a sequence of optimizing transformations does not change a program's behaviour. Some transformations can be applied by looking at a single function (these are called *intraprocedural optimizations*), but for others, the entire program must be analyzed (these are called *interprocedural optimizations*).

Program analysis can be done by several approches. For example, it can be done by model checking (see [19] for a survey), by abstract interpretation (see [17] for a survey) or by automated theorem proving (see [45] for a survey). The complexity of the programs that these approaches can deal with constantly increases.

In the theorem proving approach, the idea is to state the problem to solve as a theorem using the theorem prover's underlying "logic" and to prove the theorem using the tool. Several logics (or algebras) can be used as the basis of a theorem prover and this choice will have an impact in the complexity of finding a proof for a given theorem.

Kleene algebra is an interesting lightweight candidate for the automated theorem proving approach. In a nutshell, Kleene algebra is the algebraic theory of finite automata and regular expressions. More precisely, it is an axiomatic system that axiomatises equality between regular languages [25]. In the last two decades, Kleene algebra has been investigated successfully as a unifying framework for doing some *intraprocedural* program analyses [7, 18, 30, 33, 50].

Unfortunately, Kleene algebra alone is not well suited to do *interprocedural* program analyses, where the power of context-free languages is often needed to represent the control flow of a program. This dissertation is about extending Kleene algebra to allow one to reason over interprocedural programs.

**My Thesis.** My thesis is that interprocedural program analyses like formal verification and verification of compiler optimizations can be done in a Kleene-like algebraic formalism such that the analyses can be automated.

This dissertation develops an algebraic formalism that I call *visibly pushdown Kleene algebra*, and provides evidence that this formalism supports the thesis. In particular, I show that the formalism can be used in the above-mentioned interprocedural program analyses and that these analyses can be automated. The axiomatic system underlying the proposed formalism is linked with the notion of state transitions, uses equational reasoning and is able to represent well-known constructs of programming languages (sequences, alternatives, loops and code blocks) in a natural way. The generated proofs are formal, equational proofs similar to the mathematical style that is learned in high school.

Visibly pushdown Kleene algebra extends Kleene algebra by adding a family of implicit least fixed point operators based on a restriction of context-free grammars. In fact, visibly pushdown Kleene algebra axiomatises exactly the equational theory of

visibly pushdown languages. Visibly pushdown languages are a subclass of context-free languages defined by Alur and Madhusudan in the model checking framework to model check interprocedural programs while remaining decidable [1]. The resulting complexity of the equational theory of visibly pushdown Kleene algebra is EXPTIME-complete whereas that of Kleene algebra is PSPACE-complete.

## 1.1 Contributions

This dissertation contributes to the state-of-the-art both in interprocedural program analyses and in Kleene-like algebraic formalisms. In particular, this dissertation introduces the following results.

1. The development of an example (a "model") of the proposed algebraic formalism. This model is based on a language theory used in model checking that allows one to represent executions of interprocedural programs.

2. The development of an original algebraic formalism and the definition of syntactic sugar for the formalism to ease its use in program analysis.

3. A proof that the equations that can be proved in this algebraic formalism matches exactly the equations that are satisfied in the intended model of this formalism. So, the formalism is a faithful abstraction of the model and one can use its intuition about the model to infer some results in the formalism.

4. A proof that one can determine in EXPTIME if an equation can be proved in a large part of the proposed algebraic formalism. Also, it is shown that this is a lower bound since this problem is EXPTIME-hard (this is related to the language equivalence problem of visibly pushdown automata [1]). This is an important step for the automation of the interprocedural program analyses using this formalism.

5. The definition of two interprocedural program analyses, namely formal verification and verification of compiler optimizations, using the proposed algebraic formalism. Several examples of these analyses are also presented.

All results presented in this dissertation are personal unless stated otherwise.

## 1.2 Outline of the Dissertation

This dissertation is organized to provide evidence that support the thesis. The basics to develop the proposed formalism are introduced in Chapter 2. These basics include the presentation of an algebraic formalism, namely Kleene algebra, and the definition of a superset of regular languages called visibly pushdown languages, borrowed from the model checking community. Then, the proposed formalism to support the thesis is partially defined in Chapter 3 (and the proof of a key theorem of this chapter is given in Appendix A). In fact, a large part of the formalism is introduced in Chapter 3. The suitability of this part for the automation of a proof existence procedure is analyzed in Chapter 4 (and the proof that this procedure works takes place in Appendices B to E). In particular, a connection between the formalism and its intended model is defined. Next, the definition of the proposed formalism is completed in Chapter 5 and syntactic sugar is also added to the algebraic formalism to ease its use in program analysis. The proof of some key lemmas of this chapter are proved in Appendix F. Afterward, two interprocedural program analyses, namely formal verification and verification of compiler optimizations, are defined in Chapter 6 using the proposed formalism. Several examples of these analyses are also introduced. In conclusion, a review of all the evidence presented in this dissertation is done in Chapter 7 and open problems and future work are identified.

# Chapter 2

# Basics

This chapter briefly reviews an algebraic formalism, namely Kleene algebra, that allows one to do *intraprocedural* program analyses. Then, the weaknesses of Kleene algebra, based on regular languages, to represent *interprocedural* program analyses are identified. This chapter culminates by presenting a superset of regular languages called visibly pushdown languages, borrowed from the model checking community, that allows one to do formal verification of interprocedural programs, but that does not have (currently) a related algebraic formalism. The proposed algebraic formalism developed in this thesis is based on visibly pushdown languages.

## 2.1 Kleene Algebra (KA)

Kleene algebra is the algebraic theory of finite automata and regular expressions. Finite automata and regular expressions come from an article of Stephen Cole Kleene in the 1950s [24]. In this article, Kleene defined regular languages as the languages that are accepted by finite automata. He also showed that regular languages are exactly the languages denoted by regular expressions. Since then, regular expressions and finite automata have had a great impact in several fields of computer science.

**Definition 2.1** (Regular expressions)**.** The class of regular expressions is the smallest class of expressions on an alphabet $\Sigma$ that contains each element of $\Sigma$, the expressions 0 and 1, and that is closed under the binary operations $+$ and $\cdot$ and unary operation $^*$. The language denoted by a regular expression $p$ is noted $\mathcal{L}(p)$ and is defined by[1]

$$\mathcal{L}(0) := \emptyset, \qquad \mathcal{L}(1) := \{\varepsilon\}, \qquad \mathcal{L}(a) := \{a\} \text{ for any } a \in \Sigma,$$

[1] The symbol $\varepsilon$ denotes the empty word.

and extends homomorphically over the structure of regular expressions where

- $\cdot$ becomes $\bullet$ (concatenation of languages);
- $+$ becomes $\cup$ (union of languages);
- $^*$ becomes the set operator $^*$ (Kleene closure of languages). ♦

Kleene algebra is an algebra that axiomatises the equality between regular expressions. In this dissertation, we use Kozen's axiomatization of Kleene algebra [25].

**Definition 2.2** (Kleene algebra [25])**.** A Kleene algebra (KA) is an algebraic structure $(K, +, \cdot, ^*, 0, 1)$ satisfying the following axioms[2].

$$\begin{array}{ccc}
p + (q + r) = (p + q) + r & p(qr) = (pq)r & p = p + 0 \\
p(q + r) = pq + pr & p + q = q + p & p0 = 0 = 0p \\
(p + q)r = pr + qr & p + p = p & p1 = p = 1p \\
qp + r \leqslant p \to q^*r \leqslant p & 1 + p^*p \leqslant p^* & p \leqslant q \leftrightarrow p + q = q \\
pq + r \leqslant p \to rq^* \leqslant p & 1 + pp^* \leqslant p^* &
\end{array}$$ ♦

The theory of Kleene algebra is very interesting in itself. For the purpose of this dissertation, we will need only a small fraction of this theory. A more complete presentation of Kleene algebra can be found in [6, 31].

The axiomatic system of Kleene algebra has several interesting models like the language-theoretic model and the matrix model. For an alphabet $\Sigma$, the language-theoretic model of Kleene algebra is the algebra with universe $2^{\Sigma^*}$ along with operations $\cup$, $\bullet$, $^*$, $\emptyset$ and $\{\varepsilon\}$. The natural interpretation for the language-theoretic model is $\mathcal{L}$, as outlined above.

It turns out that Kozen's axiomatization of Kleene algebra represents exactly the equality between regular expressions. Thus, for any expressions $p$ and $q$ of the syntax of KA [25],

$$\vdash p = q \iff \mathcal{L}(p) = \mathcal{L}(q) \ .$$

In more technical terms, the equational theory of Kleene algebra is sound and complete for the language-theoretic model under its natural interpretation. Furthermore, the problem of deciding if $p = q$ is a theorem of Kleene algebra is PSPACE-complete [25].

[2] In the sequel, we write $pq$ instead of $p \cdot q$. The increasing precedence of the operators is $+$, $\cdot$ and $^*$.

Here are some useful laws of Kleene algebra for any regular expressions $p$, $q$ and $r$:

- $p^* = 1 + pp^*$;
- $p^* = 1 + p^*p$;
- $p^* = p^*p^*$;
- $p^* = (p^*)^*$;
- $(p + q)^* = p^*(qp^*)^*$ (denesting rule);
- $p(qp)^* = (pq)^*p$ (sliding rule);
- $qp = pr \rightarrow q^*p = pr^*$ (bisimulation rule);
- $q \leqslant p \wedge r \leqslant p \leftrightarrow q + r \leqslant p$.

Note that we will use freely these laws in the remainder of this dissertation.

### 2.1.1 The Matrix Model of Kleene Algebra

We now present the matrix model for Kleene algebra that will be used to prove a completeness theorem for our proposed formalism (visibly pushdown Kleene algebra). The matrix model states that the family of matrices over a Kleene algebra again forms a Kleene algebra (see for example [16, 25]). Let $\mathcal{K}$ be a Kleene algebra. The notation $\mathbf{A}[i, j]$ refers to the entry in row $i$ and column $j$ of the matrix $\mathbf{A}$. Since it is a matrix over $\mathcal{K}$, the element $\mathbf{A}[i, j]$ is any element of the universe of $\mathcal{K}$. Let $\mathbf{0}$ be the matrix whose entries are all 0. Let $\mathbf{I}$ be the identity matrix. In other words, for any row $i$ and column $j$ of $\mathbf{I}$,

$$\mathbf{I}[i, j] := \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases}$$

Note that the elements 0 and 1 in the matrices $\mathbf{0}$ and $\mathbf{I}$ are the constants 0 and 1 of $\mathcal{K}$.

Some standard operations on matrices are defined. The matrix addition operation $+$ between matrices $\mathbf{A}$ and $\mathbf{B}$ of the same size is defined, for any entry $i, j$, by

$$(\mathbf{A} + \mathbf{B})[i, j] := \mathbf{A}[i, j] + \mathbf{B}[i, j] \ ,$$

where $+$ is taken from $\mathcal{K}$. The matrix multiplication operator $\bullet$ between matrices $\mathbf{A}$ of size $n_1 \times n_2$ and $\mathbf{B}$ of size $n_2 \times n_3$ is defined, for any entry $i, j$, by

$$(\mathbf{A} \bullet \mathbf{B})[i,j] := (\sum\ k \mid 1 \leqslant k \leqslant n_2 : \mathbf{A}[i,k] \cdot \mathbf{B}[k,j]) \ ,$$

in which the right-hand side expression is a quantification over $\sum$ (which represents the operator $+$ of $\mathcal{K}$) having quantified variable $k$, range $1 \leqslant k \leqslant n_2$ and body $\mathbf{A}[i,k] \cdot \mathbf{B}[k,j]$ where $\cdot$ is taken from $\mathcal{K}$. The matrix Kleene star operation $^\star$ for a square matrix $\mathbf{A}$ of size $n \times n$ is defined recursively. If $n = 1$, $[a]^\star := [a^*]$ where the $^*$ is taken from $\mathcal{K}$. If $n \geqslant 2$, then it is always possible to partition the matrix $\mathbf{A}$ into four nonempty submatrices such that

$$\mathbf{A} := \left[\begin{array}{c|c} \mathbf{B} & \mathbf{C} \\ \hline \mathbf{D} & \mathbf{E} \end{array}\right] \ ,$$

where $\mathbf{B}$ and $\mathbf{E}$ are square. Then,

$$\left[\begin{array}{c|c} \mathbf{B} & \mathbf{C} \\ \hline \mathbf{D} & \mathbf{E} \end{array}\right]^\star := \left[\begin{array}{c|c} \mathbf{F}^\star & \mathbf{F}^\star\mathbf{C}\mathbf{E}^\star \\ \hline \mathbf{E}^\star\mathbf{D}\mathbf{F}^\star & \mathbf{E}^\star + \mathbf{E}^\star\mathbf{D}\mathbf{F}^\star\mathbf{C}\mathbf{E}^\star \end{array}\right] \ ,$$

where $\mathbf{F} := \mathbf{B} + \mathbf{C}\mathbf{E}^\star\mathbf{D}$. It is usual to define an abbreviation for any square matrix $\mathbf{A}$:

$$\mathbf{A}^+ := \mathbf{A} \cdot \mathbf{A}^\star \ .$$

Note that the matrix model will be used freely in the remainder of the dissertation. In particular, we will use matrices in Chapter 4 to model the transition relation of automata.

### 2.1.2 Kleene Algebra with Tests (KAT)

Kleene algebra with tests is an extension of Kleene algebra proposed by Dexter Kozen in 1996 [26, 27]. This extension is useful to model and to reason about *intraprocedural* programs.

The basic idea of Kleene algebra with tests is to distinguish *test* elements from *instruction* elements in the universe of the algebra. To achieve it, a universe $B$ is extracted from the universe $K$ of a Kleene algebra and $B$ will stand as the set of possible test elements. The standard operators of Kleene algebra still work in universe $B$, but it is now required that, when all operands applied to an operator of Kleene algebra come from $B$, the result is an element of $B$. Also, the universe $B$ acts as a Boolean algebra: the operators $+$ and $\cdot$ now operate on $B$ like the disjunction and

conjunction of Boolean algebra, and 0 and 1 act as true and false values. There is also a new operator that can be applied only on tests: the complementation operator $\overline{\ }$. Here is the formal definition of Kleene algebra with tests.

**Definition 2.3** (Kleene algebra with tests)**.** A Kleene algebra with tests is an algebraic structure $(K, B, +, \cdot, ^*, \overline{\ }, 0, 1)$ such that

- $B \subseteq K$;
- $(K, +, \cdot, ^*, 0, 1)$ is a Kleene algebra;
- $(B, +, \cdot, \overline{\ }, 0, 1)$ is a Boolean algebra.

In fact, it suffices to add these two axioms to Kleene algebra to obtain the axiomatization of Kleene algebra with tests:

$$b\overline{b} = 0 \qquad \text{and} \qquad b + \overline{b} = 1 \ . \qquad \blacklozenge$$

Kleene algebra with tests has been used in several program analyses, for example in formal verification of intraprocedural programs [30], in concurrency control [15] and in verification of intraprocedural compiler optimizations [33]. The idea of all these program analyses is to encode the program to be analysed (and the property to be checked or another artifact) into Kleene algebra with tests to generate a formula such that if the formula holds in KAT (i.e., it is a theorem of KAT), then the analysis outputs "yes, the program satisfies the analysis" and, if the formula does not hold, then the analysis outputs "no, the program does not satisfy the analysis".

We will not see in this section the complete encoding in KAT for the previous program analyses since the representation of interprocedural programs in our proposed formalism (an extension of KAT) will be presented in full details in Chapter 6. Here, we only give an overview of the encoding of intraprocedural programs in Kleene algebra with tests. In program semantics, the operators of KAT applied on *instruction* elements have the intuitive meaning described in Table 2.1, and the operators applied on *test* elements have the intuitive meaning described in Table 2.2. Using this intuitive meaning, the control flow of the standard programming constructs of imperative programs is easy to encode in KAT:

$$s;t := st, \qquad \textbf{if } b \textbf{ then } s \textbf{ else } t := bs + \overline{b}t, \qquad \textbf{while } b \textbf{ do } s := (bs)^*\overline{b},$$

| Operator | Intuitive meaning in program semantics |
|---|---|
| $+$ | Non-deterministic choice |
| $\cdot$ | Sequential composition |
| $^*$ | Finite iteration |
| 0 | The abort instruction |
| 1 | The skip instruction |

Table 2.1: Intuitive meaning of the KAT's operators applied on instructions.

| Operator | Intuitive meaning in program semantics |
|---|---|
| $+$ | Disjunction |
| $\cdot$ | Conjunction |
| $\overline{\phantom{x}}$ | Negation |
| 0 | The false value |
| 1 | The true value |

Table 2.2: Intuitive meaning of the KAT's operators applied on tests.

where $b$ is a test and $s$ and $t$ are programs. The expression encoding a program is then used to generate the formula to verify. See the respective works for more details [15, 30, 33].

### 2.1.3 Weaknesses of Kleene Algebra to Represent Interprocedural Programs

Kleene algebra is the algebraic theory of finite automata and regular expressions. So, Kleene algebra is built on well-known concepts for programmers, but Kleene algebra seems to handle elegantly only *intraprocedural* programs[3] (i.e., programs without functions). Applications of Kleene algebra[4] seem to confirm this thought since Kleene algebra has been used successfully to do some intraprocedural program analyses [7, 30, 33], but it has not been used directly to deal with interprocedural programs. In particular, it seems difficult to represent recursive functions and, more importantly, the idea of

[3] In fact, these programs must be abstracted to be used with Kleene algebra since Kleene algebra alone cannot handle assignment "directly". This will also be true in our proposed formalism.

[4] In this dissertation, we sometimes use the term "Kleene algebra" as a misnomer (we should use "Kleene algebra with tests" instead). We allow it to simplify the discussion. It is not really dangerous since the two theories are strongly related.

local scope of a variable. Note that the representation of the control flow of a program may be a context-free language (which subsumes regular languages) if there are (mutually) recursive functions. So, Kleene algebra alone is not well suited to deal with interprocedural programs.

Some work has been done to extend Kleene algebra to handle subclasses of context-free languages [5, 36, 37, 39]. However, these extensions do not seem to be satisfactory for interprocedural program analyses:

(i) the complexity of the equational theory of these extensions is unknown or undecidable;

(ii) frameworks for formal verification of interprocedural programs, based on these extensions, are only able to deal with regular properties[5];

(iii) none of these extensions has been applied to the verification of interprocedural compiler optimizations.

Remark (i) is disappointing: it says that it is not easy to build an automated or semi-automated tool to do interprocedural program analyses.

Remark (ii) is disturbing: non-regular properties are interesting for formal verification of interprocedural programs. In particular, the ability to use the nesting structure of procedure calls and returns (procedural context) when defining properties is useful. For example, here are some non-regular properties[6]:

**Secure file manipulation policies** like "whenever a secret file is opened in a secure procedural context, it must be closed before control exits the current context";

**Stack-sensitive security properties** like "a program must not execute a sensitive operation at any point when an untrusted procedure is currently on the stack or has ever been on the stack";

**Logging policies** like "whenever a procedure returns an error value, the error must be logged via a log procedure before control leaves the current procedural context".

Remark (iii) is annoying: results concerning the verification of several *intraprocedural* optimizations in Kleene algebra with tests have already been obtained by Kozen and Patron [33]. However, no one extended their results to *interprocedural* optimizations.

[5] Properties described by regular sets.

[6] This list is mostly inspired by [13].

How can we address remarks (i) to (iii) in a common axiomatic system while keeping the elegance of Kleene algebra? Recall that regular expressions (and, thus, Kleene algebra) were inspired by finite automata. Once again, automata theory inspires us as we will see in the following section.

## 2.2 Visibly Pushdown Languages, a Way to Represent Executions of Interprocedural Programs

The model checking community already works on the formal verification of interprocedural programs [1, 12, 46]. However, most of the tools developed so far can only deal with regular properties. There is an exception: Alur and Madhusudan defined a subclass of context-free languages they called *visibly pushdown languages* [1].

Visibly pushdown languages are a natural candidate for the representation of executions of interprocedural programs. In fact, the metaphor at the basis of a word (a string created from an alphabet $\Sigma$) in these languages is exactly a trace of an interprocedural program. To leverage the metaphor to an interesting, but still abstract, level, the alphabet $\Sigma$ of a visibly pushdown language is divided into three disjoint sets $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ which represent, respectively, the set of internal actions, the set of calls and the set of returns of an interprocedural program. This division is similar to "typing" the elements of $\Sigma$.

The class of visibly pushdown languages is defined by acceptance by a subclass of pushdown automata called *visibly pushdown automata*. The idea behind visibly pushdown automata is to drive the stack manipulations of the automaton according to the current "type" of input symbol it reads. The class of visibly pushdown languages is surprisingly robust and the language equivalence problem is EXPTIME-complete [1]. So, it is possible to use visibly pushdown automata in model checking to represent both the (mutually) recursive program and the property to be checked. In fact, some non-regular properties like those presented in remark (ii) of page 11 can be expressed with visibly pushdown automata.

### 2.2.1 (Semi-)Visibly Pushdown Automata ((S-)VPA)

Visibly pushdown automata were introduced by Mehlhorn in 1980 under the name input driven pushdown automata [40] and reinvented by Alur and Madhusudan in 2004 [1].

Visibly pushdown automata are a particular case of pushdown automata in which the stack manipulations are driven (made "visible") by the input word, which can be thought of as a string representing an execution of an interprocedural program. To allow this, the input alphabet $\Sigma$ of a visibly pushdown automaton is divided into three disjoint sets $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ which represent, respectively, the set of internal actions, the set of calls and the set of returns of a program. The idea behind this is: when a visibly pushdown automaton reads

(i) an internal action, it cannot modify the stack;

(ii) a call action, it must push a symbol on the stack;

(iii) a return action, it must read the symbol on the top of the stack and pop it, unless it is the bottom-of-stack symbol.

This idea is quite simple but useful in program analysis: since a word is an execution of a program and the program's source code is usually available for the programmers, it is easy to infer the "type" of an action in a word.

Let $\varepsilon$ be the empty word and let the set of finite words on the alphabet $\Sigma_{\mathbf{i}} \cup \Sigma_{\mathbf{c}} \cup \Sigma_{\mathbf{r}}$ be denoted by $(\Sigma_{\mathbf{i}} \cup \Sigma_{\mathbf{c}} \cup \Sigma_{\mathbf{r}})^*$.

**Definition 2.4.** A *visibly pushdown automaton* (VPA) is a structure

$$(S, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma \cup \{\bot\}, \delta, I, F)$$

in which:

- $S$ is a finite set of states;
- $I \subseteq S$ is the set of initial states;
- $F \subseteq S$ is the set of accepting states;
- $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ are three disjoint input alphabets;
- $\Gamma$ is the stack alphabet and $\bot$ is the bottom-of-stack symbol;
- $\delta$ is a transition relation of type

$$\begin{aligned} & S \times \Sigma_{\mathbf{i}} \times \{\lambda\} \times S \times \{\lambda\} \\ \cup\ & S \times \Sigma_{\mathbf{c}} \times \{\lambda\} \times S \times \Gamma \\ \cup\ & S \times \Sigma_{\mathbf{r}} \times \Gamma \times S \times \{\lambda\} \\ \cup\ & S \times \Sigma_{\mathbf{r}} \times \{\bot\} \times S \times \{\bot\} \end{aligned}$$

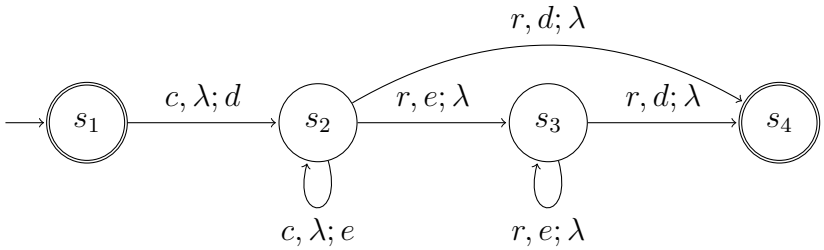
where $\lambda$ stands for "no action on the stack". A transition in this relation is represented by a quintuple $(s_1, a, d; s_2, d_2)$ in which $s_1$ is the current state, $a$ is the current input symbol read by the automaton, $d_1$ is the popped symbol, $s_2$ is the new state and $d_2$ is the pushed symbol.

A visibly pushdown automaton *accepts a word* $w \in (\Sigma_{\mathbf{i}} \cup \Sigma_{\mathbf{c}} \cup \Sigma_{\mathbf{r}})^*$ if and only if there exists at least one run of the automaton on $w$ that starts in an initial state with a stack containing only $\perp$, follows a transition for every single action of the word, and ends in an accepting state. ♦

Visibly pushdown automata define a robust class of languages called *visibly pushdown languages*. Here are some examples of visibly pushdown automata. A visibly pushdown automaton that accepts the non-regular language $\{c^n\, r^n \mid c \in \Sigma_{\mathbf{c}} \wedge r \in \Sigma_{\mathbf{r}} \wedge n \in \mathbb{N}\}$ is the structure

$$(\{s_1, s_2, s_3, s_4\}, \{\} \cup \{c\} \cup \{r\}, \{d, e\} \cup \{\perp\}, \delta, \{s_1\}, \{s_1, s_4\})$$

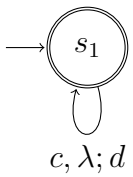
where the relation $\delta$ is represented by the following transition diagram:




In the preceding example, note that the number of calls is identical to the number of returns. This is not always the case in visibly pushdown languages. For example, the langage $\{c^n \mid c \in \Sigma_{\mathbf{c}} \wedge n \in \mathbb{N}\}$ is a visibly pushdown language because it is accepted by the following visibly pushdown automaton:

$$(\{s_1\}, \{\} \cup \{c\} \cup \{\}, \{d\} \cup \{\perp\}, \delta, \{s_1\}, \{s_1\})$$

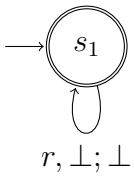
where the relation $\delta$ is represented by the following transition diagram:




The langage $\{r^n \mid r \in \Sigma_{\mathbf{r}} \wedge n \in \mathbb{N}\}$ is also a visibly pushdown language because it is accepted by the following visibly pushdown automaton:

$$(\{s_1\}, \{\} \cup \{\} \cup \{r\}, \{d\} \cup \{\perp\}, \delta, \{s_1\}, \{s_1\})$$

where the relation $\delta$ is represented by the following transition diagram:




Let $\Sigma_{\mathbf{i}} := \{\mathsf{a}\}$, $\Sigma_{\mathbf{c}} := \{\mathsf{c}\}$ and $\Sigma_{\mathbf{r}} := \{\mathsf{r}\}$. To give the reader a feeling of the class of visibly pushdown languages, here is a list of visibly pushdown languages without the proof that they really are visibly pushdown languages:

- $\{\mathsf{a}^n \mid n \in \mathbb{N}\}$ ;
- $\{\mathsf{c}^m\, \mathsf{a}^n\, \mathsf{r}^m\, \mathsf{a}\, \mathsf{a}\, \mathsf{c}^l\, \mathsf{r}^l \mid l, m, n \in \mathbb{N}\}$ ;
- $\{\mathsf{r}\, \mathsf{r}\, \mathsf{r}\, \mathsf{c}^m\, \mathsf{a}^n\, \mathsf{r}^l \mid l, m, n \in \mathbb{N} \wedge m \geqslant l\}$ .

Note that a finite word $w := \sigma_1\sigma_2\sigma_3 \ldots \sigma_n$ in $(\Sigma_{\mathbf{i}} \cup \Sigma_{\mathbf{c}} \cup \Sigma_{\mathbf{r}})^*$, where each $\sigma_i$ is a letter from $\Sigma_{\mathbf{i}} \cup \Sigma_{\mathbf{c}} \cup \Sigma_{\mathbf{r}}$, may have *pending calls* and *pending returns*. Intuitively, a pending call is a call action $\sigma_i \in \Sigma_{\mathbf{c}}$ that is not matched with a return action $\sigma_j \in \Sigma_{\mathbf{r}}$ where $i < j$, and a pending return is a return action $\sigma_i \in \Sigma_{\mathbf{r}}$ that is not matched with a call action $\sigma_j \in \Sigma_{\mathbf{c}}$ where $j < i$. For example, in the word

$$\mathsf{a}\, \mathsf{f}\rangle\, \mathsf{b}\, \mathsf{b}\, \langle \mathsf{f}\, \langle \mathsf{g}\, \mathsf{a}\, \mathsf{a}\, \mathsf{a}\, \mathsf{g}\rangle\, \mathsf{b}$$

for which $\Sigma_{\mathbf{i}} := \{\mathsf{a}, \mathsf{b}\}$, $\Sigma_{\mathbf{c}} := \{\langle \mathsf{f}, \langle \mathsf{g}\}$ and $\Sigma_{\mathbf{r}} := \{\mathsf{f}\rangle, \mathsf{g}\rangle\}$, the first action $\mathsf{f}\rangle$ is a pending return and the first action $\langle \mathsf{f}$ is a pending call. Obviously, $\langle \mathsf{g}$ and $\mathsf{g}\rangle$ are well matched since $\langle \mathsf{g}$ occurs before $\mathsf{g}\rangle$ in the word and there is no other pending call or pending return between $\langle \mathsf{g}$ and $\mathsf{g}\rangle$. Note that we used particular constants like $\langle \mathsf{f}$ and $\mathsf{f}\rangle$ for the set of call and return actions. This was done only to emphasize the structure of the word. Instead, we could have used the more classical word

$$\mathsf{a}\, \mathsf{r}\, \mathsf{b}\, \mathsf{b}\, \mathsf{c}\, \mathsf{d}\, \mathsf{a}\, \mathsf{a}\, \mathsf{a}\, \mathsf{s}\, \mathsf{b}$$

for which $\Sigma_{\mathbf{i}} := \{\mathsf{a}, \mathsf{b}\}$, $\Sigma_{\mathbf{c}} := \{\mathsf{c}, \mathsf{d}\}$ and $\Sigma_{\mathbf{r}} := \{\mathsf{r}, \mathsf{s}\}$. We will use the two styles in this dissertation with a slight preference for the former when encoding programs in our proposed formalism.

A visibly pushdown language can contain words that have pending calls and pending returns. Pending calls and pending returns are necessary to have a class of languages closed under the prefix-closure and suffix-closure operators. The number of pending

calls and pending returns in a word can be calculated by reading it left-to-right (respectively, right-to-left), counting 1 for a call (respectively, a return) if the remaining suffix (respectively, prefix) of the word does not have pending returns (respectively, pending calls), and counting 0 otherwise.

More explicitly, the operator $\mathbf{pc} : (\Sigma_\mathbf{i} \cup \Sigma_\mathbf{c} \cup \Sigma_\mathbf{r})^* \to \mathbb{N}$ that counts the number of pending calls in a word and the operator $\mathbf{pr} : (\Sigma_\mathbf{i} \cup \Sigma_\mathbf{c} \cup \Sigma_\mathbf{r})^* \to \mathbb{N}$ that counts the number of pending returns in a word are defined by mutual induction over the structure of a finite word $w \in (\Sigma_\mathbf{i} \cup \Sigma_\mathbf{c} \cup \Sigma_\mathbf{r})^*$ as follows:

$$\mathbf{pc}(w) := \begin{cases} 0 & \text{if } w \in \Sigma_\mathbf{i} \cup \Sigma_\mathbf{r} \cup \{\varepsilon\}, \\ 1 & \text{if } w \in \Sigma_\mathbf{c}, \\ 1 + \mathbf{pc}(w') & \text{if } w = \sigma w' \text{ where } \sigma \in \Sigma_\mathbf{c} \text{ and } \mathbf{pr}(w') = 0, \\ \mathbf{pc}(w') & \text{otherwise (where } w = \sigma w' \text{ with } \sigma \in \Sigma_\mathbf{i} \cup \Sigma_\mathbf{c} \cup \Sigma_\mathbf{r}), \end{cases}$$

$$\mathbf{pr}(w) := \begin{cases} 0 & \text{if } w \in \Sigma_\mathbf{i} \cup \Sigma_\mathbf{c} \cup \{\varepsilon\}, \\ 1 & \text{if } w \in \Sigma_\mathbf{r}, \\ 1 + \mathbf{pr}(w') & \text{if } w = w'\sigma \text{ where } \sigma \in \Sigma_\mathbf{r} \text{ and } \mathbf{pc}(w') = 0, \\ \mathbf{pr}(w') & \text{otherwise (where } w = w'\sigma \text{ with } \sigma \in \Sigma_\mathbf{i} \cup \Sigma_\mathbf{c} \cup \Sigma_\mathbf{r}). \end{cases}$$

A word $w \in (\Sigma_\mathbf{i} \cup \Sigma_\mathbf{c} \cup \Sigma_\mathbf{r})^*$ is said to be *well matched* if and only if it does not have pending calls or pending returns. Note that the set of all words in $(\Sigma_\mathbf{i} \cup \Sigma_\mathbf{c} \cup \Sigma_\mathbf{r})^*$ can be described easily with well-matched words, pending calls and pending returns. Let $\Sigma^{\mathsf{wm}}$ be the set of all well-matched words in $(\Sigma_\mathbf{i} \cup \Sigma_\mathbf{c} \cup \Sigma_\mathbf{r})^*$. It is easy to see that

$$(\Sigma_\mathbf{i} \cup \Sigma_\mathbf{c} \cup \Sigma_\mathbf{r})^* = (\Sigma_\mathbf{r} \cup \Sigma^{\mathsf{wm}})^* \bullet (\Sigma_\mathbf{c} \cup \Sigma^{\mathsf{wm}})^* \ , \tag{2.1}$$

or, equivalently,

$$(\Sigma_\mathbf{i} \cup \Sigma_\mathbf{c} \cup \Sigma_\mathbf{r})^* = (\Sigma_\mathbf{r} \cup \Sigma^{\mathsf{wm}})^* \bullet (\Sigma_\mathbf{c} \bullet \Sigma^{\mathsf{wm}})^* \ . \tag{2.2}$$

We prefer the latter equation (Equation (2.2)) in this dissertation. It simplifies some proofs.

The notion of the *height of a well-matched word* will be useful in this dissertation. The height of a well-matched word measures its hierarchical complexity. In other words, it gives the largest number of pending calls (or stack height) among all possible affixes of a well-matched word. For example, if $\Sigma_\mathbf{i} := \{\mathsf{a}, \mathsf{b}\}$, $\Sigma_\mathbf{c} := \{\mathsf{c}, \mathsf{d}\}$ and $\Sigma_\mathbf{r} := \{\mathsf{r}, \mathsf{s}\}$, then the well-matched word $\mathsf{acbbddaasasbbrcar}$ has a height of 3 because of the affix $\mathsf{cbbdd}$. Note that a well-matched word of height 0 is any well-matched word not containing call actions or return actions. In other words, it is a word of $\Sigma_\mathbf{i}^*$. We allow ourselves to give the height of a well-matched subword of a word.

The class of visibly pushdown languages is a strict subclass of deterministic context-free languages and a strict superclass of regular languages and balanced languages [1]. For example, we saw that $\{c^n\, r^n \mid n \in \mathbb{N} \wedge c \in \Sigma_{\mathbf{c}} \wedge r \in \Sigma_{\mathbf{r}}\}$ and $\{c^n \mid n \in \mathbb{N} \wedge c \in \Sigma_{\mathbf{c}}\}$ are VPLs, but the language $\{a^n\, r^n \mid n \in \mathbb{N} \wedge a \in \Sigma_{\mathbf{i}} \wedge r \in \Sigma_{\mathbf{r}}\}$ is not since the stack cannot be used when reading internal actions.

Visibly pushdown languages are surprisingly robust: they are closed under union, concatenation, Kleene star, intersection, complementation and prefix-closure [1, 2]. In Section 4.3, we will show some of these results algebraically.

A very interesting fact about VPLs is that their language equivalence problem is EXPTIME-complete [1]. Recall that the language equivalence problem is undecidable for context-free languages and PSPACE-complete for regular languages.

Deterministic visibly pushdown languages are as expressive as nondeterministic ones. Definition 2.4 defines what is sometimes called a *nondeterministic visibly pushdown automaton*, but it is also possible to define *deterministic* VPAs.

**Definition 2.5.** A *deterministic* visibly pushdown automaton is a visibly pushdown automaton

$$(S, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma \cup \{\bot\}, \delta, I, F)$$

in which:

- there is only one initial state;
- for every $s \in S$ and $a \in \Sigma_{\mathbf{i}}$, there exists one and only one $s' \in S$ such that $(s, a, \lambda; s', \lambda) \in \delta$;
- for every $s \in S$ and $c \in \Sigma_{\mathbf{c}}$, there exists one and only one $s' \in S$ and $d \in \Gamma$ such that $(s, c, \lambda; s', d) \in \delta$;
- for every $s \in S$, $r \in \Sigma_{\mathbf{r}}$ and $d \in \Gamma$, there exists one and only one $s' \in S$ such that $(s, r, d; s', \lambda) \in \delta$;
- for every $s \in S$, $r \in \Sigma_{\mathbf{r}}$, there exists one and only one $s' \in S$ such that $(s, r, \bot; s', \bot) \in \delta$. ♦

Nondeterministic VPAs can be determinized as we will show in Section 4.3. Such a result does not hold for general pushdown automata.

It is not difficult to see that the definition of visibly pushdown automata can be relaxed a bit to allow *some* $\varepsilon$-transitions while still accepting the same class of languages.

For example, it can be relaxed to allow an $\varepsilon$-transition to change the current top of the stack symbol into the bottom-of-stack symbol (and thus blocking up the section of the stack below this symbol). This relaxation gives rise to the concept of *semi-visibly pushdown automaton*, which is useful for the design of efficient algorithms for operations like concatenation and Kleene star on these automata. To our knowledge, the concept of semi-visibly pushdown automaton has not been defined elsewhere before.

**Definition 2.6.** A *semi-visibly pushdown automaton* (S-VPA) is a structure

$$(S, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma \cup \{\bot\}, \delta, I, F)$$

in which $S$, $I$, $F$, $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$, $\Sigma_{\mathbf{r}}$ and $\Gamma$ are as for a VPA and $\delta$ is a transition relation of type

$$\begin{aligned} & S \times \Sigma_{\mathbf{i}} \times \{\lambda\} \times S \times \{\lambda\} \\ \cup\ & S \times \Sigma_{\mathbf{c}} \times \{\lambda\} \times S \times \Gamma \\ \cup\ & S \times \Sigma_{\mathbf{r}} \times \Gamma \times S \times \{\lambda\} \\ \cup\ & S \times \Sigma_{\mathbf{r}} \times \{\bot\} \times S \times \{\bot\} \\ \cup\ & S \times \{\varepsilon\} \times (\Gamma \cup \{\bot\}) \times S \times \{\bot\} \end{aligned}$$

where $\lambda$ stands for "no action on the stack" and the following property is satisfied: for all $s, s' \in S$ and $d \in \Gamma$,

$$(s, \varepsilon, d; s', \bot) \in \delta \iff (\forall\ d' \mid d' \in \Gamma : (s, \varepsilon, d'; s', \bot) \in \delta)\ . \tag{2.3}$$

The criterion for accepting a word is the same as for a VPA. ♦

It is not difficult (but lengthy) to prove that the class of languages represented by semi-visibly pushdown automata is exactly that of visibly pushdown languages[7] [8]. The concept of semi-visibly pushdown automaton will be used only in the completeness proof of Section 4.3 where it allows us to avoid an exponential blow-up in the construction made in the proof.

As an historical note, Alur and Madhusudan were not the first to think of "typing" the atomic elements of pushdown automata. In 1980, Mehlhorn studied input driven (real-time) pushdown automata [40] which are very similar to visibly pushdown automata. Later, in 1983, Braunmühl and Verbeek further studied these input driven pushdown automata [49]. However, neither seems to have used it in formal verification of interprocedural programs.

[7] This result is proved algebraically in Theorem 4.7. In short, the proof is only an elimination of $\varepsilon$-transitions.

### 2.2.2 Visibly Pushdown Grammars

Visibly pushdown grammars were defined by Alur and Madhusudan [2]. It is a restriction of context-free grammars that characterizes exactly the visibly pushdown languages. The notion of visibly pushdown grammar inspired us for the definition of the terms of the proposed algebra.

**Definition 2.7** (Visibly pushdown grammars [2])**.** A *visibly pushdown grammar* on $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ is a structure

$$(V_0, V_1, Z, \{\rightarrow\})$$

in which:

- $V_0$ and $V_1$ are two disjoint sets of nonterminals that represent, respectively, the set of nonterminals that can only derive well-matched words and the set of nonterminals that can derive words having pending calls and pending returns;
- $Z \in V_0 \cup V_1$ is the starting nonterminal;
- $\{\rightarrow\}$ is the finite set of rewrite rules (or productions). Each rewrite rule must have one of the following forms:
  - $X \rightarrow \varepsilon$ where $X \in V_0 \cup V_1$;
  - $X \rightarrow a\,Y$ where $X, Y \in V_0$ and $a \in \Sigma_{\mathbf{i}}$;
  - $X \rightarrow c\,Y\,r\,W$ where $X, Y, W \in V_0$, $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$;
  - $X \rightarrow a\,Y$ where $X \in V_1$, $Y \in V_0 \cup V_1$ and $a \in \Sigma_{\mathbf{i}} \cup \Sigma_{\mathbf{c}} \cup \Sigma_{\mathbf{r}}$;
  - $X \rightarrow c\,Y\,r\,W$ where $X \in V_1$, $Y \in V_0$, $W \in V_0 \cup V_1$, $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$.

The semantics of a visibly pushdown grammar (and any grammar presented in this dissertation) is the same as for context-free grammars. It is defined by a derivation relation $\Rightarrow^*$ over the nonterminals and the terminals. For every rewrite rule $X \rightarrow \alpha$ of the grammar and for all "words" $\beta$ and $\gamma$ that can contain terminals and nonterminals, the relation $\beta\,X\,\gamma \Rightarrow \beta\,\alpha\,\gamma$ holds. The relation $\Rightarrow^*$ is the least fixed point of $\Rightarrow$. The *language of a grammar* having starting nonterminal $S$ is the set of all words $w$ that contain only terminals such that $S \Rightarrow^* w$ holds. In other words, $w$ is a word of the language of a grammar if and only if it can be derived from the starting nonterminal in a finite number of steps. ♦

Here is an example of a visibly pushdown grammar that generates the language $\{c^n\,r^n \mid n \in \mathbb{N} \wedge c \in \Sigma_{\mathbf{c}} \wedge r \in \Sigma_{\mathbf{r}}\}$:

$$(\{X, Y\}, \emptyset, X, \{X \rightarrow c\,X\,r\,Y,\ X \rightarrow \varepsilon,\ Y \rightarrow \varepsilon\})\ .$$

Here is another example of a visibly pushdown grammar that generates the language $\{r^m\, c^n\, r^n \mid m \geqslant 1 \wedge n \in \mathbb{N} \wedge c \in \Sigma_{\mathbf{c}} \wedge r \in \Sigma_{\mathbf{r}}\}$:

$$(\{X, Y\}, \{Z\}, Z, \{Z \to r\, Z,\ Z \to r\, X,\ X \to c\, X\, r\, Y,\ X \to \varepsilon,\ Y \to \varepsilon\})\ .$$

An interesting fact about visibly pushdown grammars is that they generate exactly the visibly pushdown languages as shown by the following theorem.

**Theorem 2.8** (Visibly pushdown grammars generate exactly the visibly pushdown languages [2])**.** *Let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be three disjoint alphabets. A language $L \subseteq (\Sigma_{\mathbf{i}} \cup \Sigma_{\mathbf{c}} \cup \Sigma_{\mathbf{r}})^*$ is a visibly pushdown language if and only if there exists a visibly pushdown grammar that generates $L$.*

It is easy to see that any visibly pushdown grammar on $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ is also a context-free grammar on $\Sigma_{\mathbf{i}} \cup \Sigma_{\mathbf{c}} \cup \Sigma_{\mathbf{r}}$, but the opposite is not true (after all, the context-free language $\{a^n\, b^n \mid n \in \mathbb{N} \wedge a, b \in \Sigma_{\mathbf{i}}\}$ is not a visibly pushdown language). Thus, visibly pushdown grammars are a strict subset of context-free grammars.

The definition of visibly pushdown grammars is linked to the notion of a well-matched word (the set $V_0$ represents the nonterminals that can generate only well-matched words). Note also that the rewrite rules having a nonterminal from $V_0$ in the left-hand side are independent from the other rewrite rules in the sense that no element of $V_0$ can generate a nonterminal from $V_1$. So, it is possible to calculate the language generated by a visibly pushdown grammar by first generating all well-matched words of $V_0$ and then generate the other words. We will use this idea in the proposed algebra when defining the terms of the algebra.

# Chapter 3

# Visibly Pushdown Kleene Algebra (VPKA)

This chapter presents the proposed formalism to support the thesis: *visibly pushdown Kleene algebra* (VPKA). In a nutshell, VPKA characterizes exactly the equality between two visibly pushdown languages. Before showing the axiomatic system, we present the terms of VPKA: *visibly pushdown regular expressions*.

## 3.1 Visibly Pushdown Regular Expressions (VPRE)

To our knowledge, Pitcher is the only one that has previously defined "expressions" (that he calls "visibly pushdown expressions") that denote exactly the visibly pushdown languages [43]. He defined these expressions in the context of XML stream processing. Visibly pushdown expressions are based on regular expression types [22]. Unfortunately, visibly pushdown expressions do not seem well suited to be the basis of a Kleene-like algebra. In particular, they have some operations which will complicate the resulting algebra. For instance, Pitcher has an intersection operator, two different concatenations operators and explicit variables (with special constraints to prevent inadequate use of these variables) that must have an algebraic definition. Moreover, it seems difficult to define mutually recursive programs in this setting.

Thus, we define our own concept of "visibly pushdown regular expressions" that denote exactly the visibly pushdown languages while remaining close to the terms of Kleene algebra. Some set operations on visibly pushdown languages are first introduced.

Let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be three disjoint finite sets. Let $S, T \subseteq (\Sigma_{\mathbf{i}} \cup \Sigma_{\mathbf{c}} \cup \Sigma_{\mathbf{r}})^*$. The concatenation operation on sets of words is defined as usual:

$$S \bullet T := \{st \mid s \in S \wedge t \in T\} \ .$$

The power $S^n$ with respect to $\bullet$ is defined inductively by $S^0 := \{\varepsilon\}$ and $S^{n+1} := S \bullet S^n$. This allows us to define the Kleene star operator by

$$S^* := (\cup\, n \mid n \in \mathbb{N} : S^n) \ ,$$

which is a quantification with quantifier $\cup$, quantified variable $n$, range $n \in \mathbb{N}$ and body $S^n$.

The standard operators of regular expressions are not sufficient to generate every visibly pushdown language from $\emptyset$, $\{\varepsilon\}$, and the singletons $\{a\}$ for $a \in \Sigma_{\mathbf{i}} \cup \Sigma_{\mathbf{c}} \cup \Sigma_{\mathbf{r}}$. So, other operators are needed. It is not difficult to see that visibly pushdown languages differ from regular languages mostly for their well-matched words. More precisely, if we are able to generate every visibly pushdown language that contains only well-matched words, we can use the standard operators of Kleene algebra on these languages and the singletons $\{a\}$ for $a \in \Sigma_{\mathbf{c}} \cup \Sigma_{\mathbf{r}}$ to generate every visibly pushdown language. So, a way is needed to generate visibly pushdown languages that contain only well-matched words. An infinite family of operators for doing that is defined. To ease the understanding of this family of operators, we give two alternative definitions of VPRE: one is grammar-based [10] and the other is "block"-based [9].

### 3.1.1 Grammar-Based Definition of Visibly Pushdown Regular Expressions

This definition is based on the idea that visibly pushdown languages that contain only well-matched words are better described via context-free grammars. Thus, in this definition, visibly pushdown regular expressions are an extension of regular expressions by a restricted set of context-free grammars. In these grammars, we will use nonterminals of the form $P_{(x,y)}$ to enforce the definition of a "starting element" $x$ and an "ending element" $y$ (the goal to reach) in each nonterminal. Here is the formal definition of these grammars.

**Definition 3.1** (Well-matched visibly pushdown grammar (WMVPG))**.** Let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be disjoint finite sets of atomic elements. Let $V$ be a finite set of symbols (or labels) containing symbols $s$ and $t$, and let $N(V) := \{P_{(x,y)} \mid x, y \in V\}$. A *well-matched visibly pushdown grammar* over $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ is a tuple $G := (V, P_{(s,t)}, \rightarrow)$ where $N(V)$ is the

set of nonterminals, $P_{(s,t)} \in N(V)$ is the starting nonterminal and $\to$ is a finite set of *explicit rewrite rules* of the form

- $P_{(x,y)} \to \varepsilon$, where $x, y \in V$;
- $P_{(x,y)} \to a$, where $a \in \Sigma_{\mathbf{i}}$ and $x, y \in V$;
- $P_{(x,y)} \to c\, P_{(z,w)}\, r$, where $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $w, x, y, z \in V$

and *implicit rewrite rules*

- $P_{(x,y)} \to P_{(x,z)}\, P_{(z,y)}$ for each $x, y, z \in V$.

The language generated by $G$ is the set of words (remark: well-matched words, in fact) that can be derived by the rewrite rules of $G$ when starting with $P_{(s,t)}$. ♦

We will use WMVPGs in set expressions. For example, for a WMVPG $G$ and elements $a_1, a_2 \in \Sigma_{\mathbf{i}}$, $\{a_1\} \bullet G \cup \{a_2\}$ is such an expression. Note that one only needs to know the explicit rewrite rules of $G$ along with its starting nonterminal $P_{(s,t)}$ since the other components of $G$ can be inferred from them. For example, a WMVPG generating $\{c^n r^n \mid n \in \mathbb{N}\}$ for $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$ can be expressed shortly by

$$(P_{(\mathsf{x},\mathsf{y})}, \{P_{(\mathsf{x},\mathsf{y})} \to c\, P_{(\mathsf{x},\mathsf{y})}\, r,\ P_{(\mathsf{x},\mathsf{y})} \to \varepsilon\})$$

and represents the WMVPG $(\{\mathsf{x}, \mathsf{y}\}, P_{(\mathsf{x},\mathsf{y})}, \to)$ where $\to$ is

$$\begin{aligned}\{&P_{(\mathsf{x},\mathsf{y})} \to c\, P_{(\mathsf{x},\mathsf{y})}\, r,\ P_{(\mathsf{x},\mathsf{y})} \to \varepsilon,\ P_{(\mathsf{x},\mathsf{x})} \to P_{(\mathsf{x},\mathsf{x})}\, P_{(\mathsf{x},\mathsf{x})},\ P_{(\mathsf{x},\mathsf{x})} \to P_{(\mathsf{x},\mathsf{y})}\, P_{(\mathsf{y},\mathsf{x})},\\ &P_{(\mathsf{x},\mathsf{y})} \to P_{(\mathsf{x},\mathsf{x})}\, P_{(\mathsf{x},\mathsf{y})},\ P_{(\mathsf{x},\mathsf{y})} \to P_{(\mathsf{x},\mathsf{y})}\, P_{(\mathsf{y},\mathsf{y})},\ P_{(\mathsf{y},\mathsf{x})} \to P_{(\mathsf{y},\mathsf{x})}\, P_{(\mathsf{x},\mathsf{x})},\\ &P_{(\mathsf{y},\mathsf{x})} \to P_{(\mathsf{y},\mathsf{y})}\, P_{(\mathsf{y},\mathsf{x})},\ P_{(\mathsf{y},\mathsf{y})} \to P_{(\mathsf{y},\mathsf{x})}\, P_{(\mathsf{x},\mathsf{y})},\ P_{(\mathsf{y},\mathsf{y})} \to P_{(\mathsf{y},\mathsf{y})}\, P_{(\mathsf{y},\mathsf{y})}\}\ .\end{aligned}$$

To ease the writing of lengthy grammars, we define a notation called the "block" notation for WMVPGs that not only shows the structure of the explicit rewrite rules in the context of a program but also emphasizes the starting nonterminal. Let $G := (V, P_{(s,t)}, \to)$ be a WMVPG. We write

$$\begin{bmatrix} \varepsilon \end{bmatrix}^{y}_{x}, \qquad \begin{bmatrix} a \end{bmatrix}^{y}_{x}, \qquad \begin{bmatrix} c \downarrow^{w}_{z} \uparrow r \end{bmatrix}^{y}_{x}$$

to respectively represent the explicit rewrite rules

$$P_{(x,y)} \to \varepsilon, \qquad P_{(x,y)} \to a, \qquad P_{(x,y)} \to c\, P_{(z,w)}\, r\ .$$

We call *unary block* each rule of the form $[_x\, \varepsilon\, ]^y$ or $[_x\, a\, ]^y$. The labels $x$ and $y$ are respectively called the *starting label* and the *ending label* of the block. We also call *binary block* each rule of the form $[_x\, c \downarrow_z \uparrow^w r\, ]^y$. The labels $x$, $y$, $z$ and $w$ are respectively called the *starting label*, the *ending label*, the *call label* and the *return label*.

Let $\mathcal{B}$ be the block notation of the explicit rewrite rules of $G$. Define $\mathcal{B}^1$ as the set of unary blocks and $\mathcal{B}^2$ as the set of binary blocks. The labels of the starting nonterminal $P_{(s,t)}$ are used to surround $\mathcal{B}$ by writing $(\!|_s\, \mathcal{B}\, |\!)^t$. Thus, $G$ is abbreviated as $(\!|_s\, \mathcal{B}\, |\!)^t$.

Here are some examples of languages generated by WMVPGs, using the block notation. Let $\Sigma_{\mathbf{i}} := \{\mathsf{a}, \mathsf{b}\}$, $\Sigma_{\mathbf{c}} := \{\mathsf{c}, \mathsf{d}\}$, $\Sigma_{\mathbf{r}} := \{\mathsf{r}, \mathsf{s}\}$ and $V := \{\mathsf{v}, \mathsf{w}, \mathsf{x}, \mathsf{y}, \mathsf{z}\}$. Then,

- $(\!|_{\mathsf{x}\,\mathsf{x}}\, [_{\mathsf{x}}\, \mathsf{a}\, ]^{\mathsf{x}\,\mathsf{x}}\, |\!) = \{\mathsf{a}^n \mid n > 0\}$;
- $(\!|_{\mathsf{x}\,\mathsf{x}}\, [_{\mathsf{x}}\, \mathsf{a}\, ]^{\mathsf{y}},\ [_{\mathsf{x}}\, \mathsf{a}\, ]^{\mathsf{y}\,\mathsf{x}}\, |\!) = \emptyset$;
- $(\!|_{\mathsf{x}\,\mathsf{x}}\, [_{\mathsf{x}}\, \mathsf{c} \downarrow \uparrow^{\mathsf{y}}\, \mathsf{r}\, ]^{\mathsf{y}},\ [_{\mathsf{x}}\, \varepsilon\, ]^{\mathsf{y}\,\mathsf{y}}\, |\!) = \{\mathsf{c}^n\, \mathsf{r}^n \mid n \in \mathbb{N}\}$;
- $(\!|_{\mathsf{x}\,\mathsf{y}}\, [_{\mathsf{v}}\, \mathsf{c} \downarrow \uparrow^{\mathsf{w}}\, \mathsf{r}\, ]^{\mathsf{z}},\ [_{\mathsf{y}}\, \mathsf{b}\, ]^{\mathsf{z}},\ [_{\mathsf{v}}\, \mathsf{d} \downarrow_{\mathsf{x}} \uparrow^{\mathsf{z}}\, \mathsf{s}\, ]^{\mathsf{w}},\ [_{\mathsf{x}}\, \mathsf{a}\, ]^{\mathsf{y}\,\mathsf{z}}\, |\!) = \{\mathsf{a}\,(\mathsf{c}\,\mathsf{d}\,\mathsf{a})^n\,\mathsf{b}\,(\mathsf{s}\,\mathsf{r})^n \mid n \in \mathbb{N}\}$.

Note from the examples that, for convenience, we write the blocks as a list rather than as a set. However, this is a set.

We are now ready to define *visibly pushdown regular expressions*.

**Definition 3.2** (Visibly pushdown regular expressions (grammar-based))**.** Let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be disjoint finite sets of atomic elements. Define a *grammar pattern* of a WMVPG $G$ on $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ to be a partial operator obtained by

- replacing the terminals in $G$ by variables (placeholders) of the same type ($\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ or $\Sigma_{\mathbf{r}}$);
- writing $[_x\, 1\, ]^y$ instead of $[_x\, \varepsilon\, ]^y$;
- adding, for convenience, blocks of the form $[_x\, 0\, ]^y$.

The arity of a grammar pattern is the number of different variables it contains. Define $\mathcal{G}$ as the set of all grammar patterns on $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$. A VPRE is any well-formed

expression that can be generated from the base elements 0, 1, $a$ for each $a \in \Sigma_{\mathbf{i}} \cup \Sigma_{\mathbf{c}} \cup \Sigma_{\mathbf{r}}$, the unary operator $^*$, the binary operators $\cdot$ and $+$, and the grammar patterns in $\mathcal{G}$. The language denoted by a VPRE $p$ is noted $\mathcal{L}(p)$ and is defined by

$$\mathcal{L}(0) := \emptyset, \qquad \mathcal{L}(1) := \{\varepsilon\}, \qquad \mathcal{L}(a) := \{a\} \text{ for any } a \in \Sigma_{\mathbf{i}} \cup \Sigma_{\mathbf{c}} \cup \Sigma_{\mathbf{r}},$$

and extends over the structure of VPREs with $\cdot$ becoming $\bullet$, $+$ becoming $\cup$, and $^*$ becoming the set operator $^*$. An operator of $\mathcal{G}$ along with its operands becomes a WMVPG $G$, and so $\mathcal{L}(G)$ is the language generated by $G$.[1] ♦

### Some Alternative Definitions of WMVPGs

The definition of WMVPGs is interesting. For one thing, nonterminals of the form $P_{(x,y)}$ are used in WMVPGs to enforce the definition of a "starting element" $x$ and an "ending element" $y$ for each nonterminal. Note how useful this is when defining the "cut" in implicit rewrite rules. Note also that the implicit rewrite rules allow multiple choices for a derivation of a word. We will see in Chapter 6 that multiple choices for a derivation are an essential ingredient to ease program manipulations.

On the other hand, one can sometimes be interested in using a *fixed* strategy (restricting the choices) when deriving nonterminals of a WMVPG. Such strategies will be useful when defining axioms for our proposed formalism. Among the possibilities to restrict the choices, one can always use an explicit rewrite rule to derive the *first* nonterminal generated by an implicit rewrite rule. Intuitively, this strategy would be the same as defining the implicit rewrite rules in WMVPGs by $P_{(x,z)} \to q\, P_{(y,z)}$ for each $x, y, z \in V$ and $q \in \{\varepsilon\} \cup \Sigma_{\mathbf{i}} \cup \Sigma_{\mathbf{c}} \times N(V) \times \Sigma_{\mathbf{r}}$ whenever the explicit rewrite rule $P_{(x,y)} \to q$ exists. It turns out that such a strategy for WMVPGs does not limit the generative capacity of the grammar as we will see below. First, let us define a new class of grammars that use this strategy and then show that it generates the same class of languages as WMVPGs.

**Definition 3.3** (Grammars restricted to well-matched VPLs using a forward strategy)**.** Let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be disjoint finite sets of atomic elements. Let $V$ be a finite set of symbols containing $s$ and $t$, and let $N(V) := \{P_{(x,y)} \mid x, y \in V\}$. A *grammar restricted to well-matched* VPL*s using a forward strategy* over $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ is a tuple $G := (V, P_{(s,t)}, \to)$ where $N(V)$ is the set of nonterminals, $P_{(s,t)} \in N(V)$ is the starting nonterminal and $\to$ is a finite set of *explicit rewrite rules* of the form

[1] Blocks of the form $[_x\, 0\, ]^y$ do not translate to a rule. They are just omitted in the grammar. Those blocks are a way to identify the `fail` instruction in programs.

- $P_{(x,y)} \to \varepsilon$ where $x, y \in V$;
- $P_{(x,y)} \to a$ where $a \in \Sigma_{\mathbf{i}}$ and $x, y \in V$;
- $P_{(x,y)} \to c\, P_{(z,w)}\, r$ where $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $w, x, y, z \in V$

and the set of *implicit rewrite rules*:

- $P_{(x,z)} \to P_{(y,z)}$ for each $x, y \in V$ where the rule $P_{(x,y)} \to \varepsilon$ is in the explicit rewrite rules and for all $z \in V$;
- $P_{(x,z)} \to a\, P_{(y,z)}$ for each $x, y \in V$ and $a \in \Sigma_{\mathbf{i}}$ where the rule $P_{(x,y)} \to a$ is in the explicit rewrite rules and for all $z \in V$;
- $P_{(x,z)} \to c\, P_{(w,w_2)}\, r\, P_{(y,z)}$ for each $x, y, w, w_2 \in V$, $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$ where the rule $P_{(x,y)} \to c\, P_{(w,w_2)}\, r$ is in the explicit rewrite rules and for all $z \in V$. ♦

Note that every grammar restricted to well-matched VPLs using a forward strategy can be associated uniquely to a $(\!|\,|\!)$-expression if we use the same abbreviations as for WMVPGs.

**Theorem 3.4.** *Grammars restricted to well-matched* VPL*s using a forward strategy generate the same class of languages as* WMVPG*s.*

*Proof.* Note that every grammar restricted to well-matched VPLs using a forward strategy is associated uniquely to a WMVPG and vice versa. Note also that every word generated by a grammar restricted to well-matched VPLs using a forward strategy can be generated by its associated WMVPG. So, it suffices to prove that every word generated by a WMVPG $G$ can also be generated by its associated grammar restricted to well-matched VPLs using a forward strategy (call it $G'$). To do it, we use the notion of the height of a well-matched word defined in Section 2.2.1. We show that for all $n \in \mathbb{N}$ and $x, y \in V$, every well-matched word $w$ of height $n$ generated by the rewrite rules of $G$ when starting in $P_{(x,y)}$ can also be generated by the rewrite rules of $G'$ when starting in $P_{(x,y)}$. We prove this by generalized induction over $n$.

For the base case $n = 0$, the well-matched word $w$ is just a sequence of internal actions. Take any derivation tree that generates $w$ using the rewrite rules of $G$ and starting with $P_{(x,y)}$. By the definition of WMVPG, the parent node of any leaf of the tree (an internal action or the empty word) comes from the explicit rewrite rules $P_{(u,u')} \to \varepsilon$ or $P_{(u,u')} \to a$ for $a \in \Sigma_{\mathbf{i}}$ and $u, u' \in V$. Moreover, by the definition of

WMVPG, it is easy to see that, when reading the tree left-to-right, the sequence of nonterminals of the explicit rewrite rules follow the pattern:

$$P_{(u_1,u_2)}\, P_{(u_2,u_3)}\, P_{(u_3,u_4)} \cdots P_{(u_{k-1},u_k)}\, P_{(u_k,u_{k+1})} \ ,$$

in which $u_i \in V$ for all $i \in \{1, 2, \ldots, k+1\}$, and, of course, $u_1 = x$ and $u_{k+1} = y$. Using this sequence, we can construct another valid derivation tree that uses only the rewrite rules of $G'$. For the case where there is only one nonterminal in the sequence, this is trivial. For the other cases, we construct the tree from bottom to top. We start with the last two nonterminals ($P_{(u_{k-1},u_k)}$ and $P_{(u_k,u_{k+1})}$). Take the leaf $m \in \{\varepsilon\} \cup \Sigma_{\mathbf{i}}$ associated to $P_{(u_{k-1},u_k)}$. Note that the rule $P_{(u_{k-1},u_{k+1})} \to m\, P_{(u_k,u_{k+1})}$ exists. Then, we consider the nonterminal $P_{(u_{k-2},u_{k-1})}$ and its associated leaf $m_2 \in \{\varepsilon\} \cup \Sigma_{\mathbf{i}}$. Again, by Definition 3.3, the rewrite rule $P_{(u_{k-2},u_{k+1})} \to m_2\, P_{(u_{k-1},u_{k+1})}$ exists. This idea can be reused for any $P_{(u_i,u_{i+1})}$ in the sequence by reading them right-to-left. Hence, we have created a valid derivation tree for the word $w$ using the rewrite rules of $G'$ and starting with $P_{(u_1,u_{k+1})}$ (and considering that $u_1 = x$ and $u_{k+1} = y$).

For the inductive case, we suppose that for all $j \in \{0, 1, \ldots, l\}$ and $x, y \in V$, every well-matched word of height $j$ generated by the rewrite rules of $G$ when starting in $P_{(x,y)}$ can also be generated by the rewrite rules of $G'$ when starting in $P_{(x,y)}$. We prove that for all $x, y \in V$, every well-matched word $w$ of height $l+1$ generated by the rewrite rules of $G$ when starting in $P_{(x,y)}$ can also be generated by the rewrite rules of $G'$ when starting in $P_{(x,y)}$.

Take any derivation tree that generates $w$ using the rewrite rules of $G$ and starting with $P_{(x,y)}$. By the definition of WMVPG, the parent node of any leaf of the tree comes from the explicit rewrite rules of the grammar. Moreover, by the definition of WMVPG, it is easy to see that, when reading left-to-right the nonterminal of any leaf that is not a return action (so it is either an internal action, a call action or the empty word), and omitting the nonterminals that have as ancestor a nonterminal coming from an explicit rewrite rule of the form $P_{(u,u')} \to c\, P_{(z,z_2)}\, r$, the sequence of nonterminals of the explicit rewrite rules follow the pattern:

$$P_{(u_1,u_2)}\, P_{(u_2,u_3)}\, P_{(u_3,u_4)} \cdots P_{(u_{k-1},u_k)}\, P_{(u_k,u_{(k+1)})} \ ,$$

in which $u_i \in V$ for all $i \in \{1, 2, \ldots, k+1\}$ and, of course, $u_1 = x$ and $u_{k+1} = y$. Note that the nonterminal of a rewrite rule of the form $P_{(u,u')} \to c\, P_{(z,z_2)}\, r$ appears once (not twice) in the sequence for a pair of elements $c$ and $r$. Using this sequence, we can construct another valid derivation tree that uses only the rewrite rules of $G'$. We first use the induction hypotheses. For any nonterminal $P_{(u_i,u_{i+1})}$ in the sequence that is linked to a rule of the form $P_{(u,u')} \to c\, P_{(z,z_2)}\, r$, the induction hypotheses say that we can find a valid derivation tree for $P_{(z,z_2)}$ since the height of the well-matched subword

generated by $P_{(z,z_2)}$ is smaller or equal to $l$. Then, we just use a reasoning similar to the base case and this allows us to generate a valid derivation tree for $w$ using the rewrite rules of $G'$ and starting with $P_{(u_1,u_{k+1})}$ (and considering that $u_1 = x$ and $u_{k+1} = y$). ■

Moreover, one can always use an explicit rewrite rule to derive the *last* nonterminal generated by an implicit rewrite rule. Again, this strategy for WMVPGs does not limit the generative capacity of the grammar as we will see below.

**Definition 3.5** (Grammars restricted to well-matched VPLs using a backward strategy)**.** Let $\Sigma_\mathbf{i}$, $\Sigma_\mathbf{c}$ and $\Sigma_\mathbf{r}$ be disjoint finite sets of atomic elements. Let $V$ be a finite set of symbols containing $s$ and $t$, and let $N(V) := \{P_{(x,y)} \mid x, y \in V\}$. A *grammar restricted to well-matched* VPL*s using a backward strategy* over $\Sigma_\mathbf{i}$, $\Sigma_\mathbf{c}$ and $\Sigma_\mathbf{r}$ is a tuple $G := (V, P_{(s,t)}, \to)$ where $N(V)$ is the set of nonterminals, $P_{(s,t)} \in N(V)$ is the starting nonterminal and $\to$ is a finite set of *explicit rewrite rules* of the form

- $P_{(x,y)} \to \varepsilon$ where $x, y \in V$;
- $P_{(x,y)} \to a$ where $a \in \Sigma_\mathbf{i}$ and $x, y \in V$;
- $P_{(x,y)} \to c\, P_{(z,w)}\, r$ where $c \in \Sigma_\mathbf{c}$, $r \in \Sigma_\mathbf{r}$ and $w, x, y, z \in V$

and the set of *implicit rewrite rules*:

- $P_{(x,z)} \to P_{(x,y)}$ for each $x, y, z \in V$ such that the rule $P_{(y,z)} \to \varepsilon$ exists;
- $P_{(x,z)} \to P_{(x,y)}\, a$ for each $x, y, z \in V$ and $a \in \Sigma_\mathbf{i}$ such that the rule $P_{(y,z)} \to a$ exists;
- $P_{(x,z)} \to P_{(x,y)}\, c\, P_{(w,w_2)}\, r$ for each $x, y, z, w, w_2 \in V$, $c \in \Sigma_\mathbf{c}$ and $r \in \Sigma_\mathbf{r}$ such that the rule $P_{(y,z)} \to c\, P_{(w,w_2)}\, r$ exists. ♦

**Theorem 3.6.** *Grammars restricted to well-matched* VPL*s using a backward strategy generate the same class of languages as* WMVPG*s.*

The proof of Theorem 3.6 is similar to the proof of Theorem 3.4.

WMVPGs are different from visibly pushdown grammars as defined by Alur and Madhusudan [2] and introduced in Section 2.2.2. In particular, visibly pushdown grammars can generate words having pending calls and pending returns, but WMVPGs cannot. They can only generate well-matched words. However, if the definition of visibly

pushdown grammar is restricted to use only nonterminals of $V_0$, then the restricted visibly pushdown grammars will generate only well-matched words. A natural question arising from this is: Do they represent the same class of languages? As we will show, the answer is yes.

We first define the restriction of visibly pushdown grammars to well-matched words. The idea is to use only the set of nonterminal $V_0$ of Definition 2.7.

**Definition 3.7** (Visibly pushdown grammar (VPG) restricted to well-matched VPLs)**.** Let $\Sigma_\mathbf{i}$, $\Sigma_\mathbf{c}$ and $\Sigma_\mathbf{r}$ be disjoint finite sets of atomic elements. A *visibly pushdown grammar restricted to well-matched* VPL*s* over $\Sigma_\mathbf{i}$, $\Sigma_\mathbf{c}$ and $\Sigma_\mathbf{r}$ is a tuple $G := (V_0, S, \to)$ where $V_0$ is a finite set of nonterminals, $S \in V_0$ is the starting nonterminal and $\to$ is a finite set of rewrite rules of the form

- $X \to \varepsilon$ where $X \in V_0$;
- $X \to a\,Y$ where $a \in \Sigma_\mathbf{i}$ and $X, Y \in V_0$;
- $X \to c\,Z\,r\,Y$ where $c \in \Sigma_\mathbf{c}$, $r \in \Sigma_\mathbf{r}$ and $X, Y, Z \in V_0$. ♦

**Theorem 3.8.** WMVPG*s generate the same class of languages as* VPG*s restricted to well-matched* VPL*s.*

The proof of Theorem 3.8 is in Appendix A.

VPGs restricted to well-matched VPLs can generate every well-matched VPL and only well-matched VPLs [2]. Thus, by Theorem 3.8, we have that WMVPGs can generate every well-matched VPL and only well-matched VPLs.

### Languages Denoted by Visibly Pushdown Regular Expressions

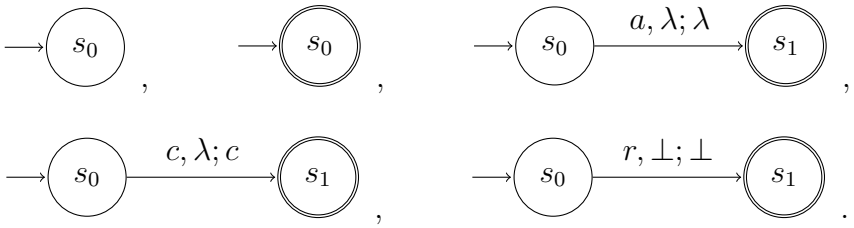
The class of visibly pushdown regular expressions is rich enough to denote exactly the visibly pushdown languages as shown by the following theorem.

**Theorem 3.9** (Theorem *à la* Kleene for visibly pushdown regular expressions)**.** *Let* $\Sigma_\mathbf{i}$, $\Sigma_\mathbf{c}$ *and* $\Sigma_\mathbf{r}$ *be three disjoint finite sets. Let* $L \subseteq (\Sigma_\mathbf{i} \cup \Sigma_\mathbf{c} \cup \Sigma_\mathbf{r})^*$ *be a language. The following propositions are equivalent:*

*(i) L is accepted by a visibly pushdown automaton;*

*(ii) $L$ is denoted by a visibly pushdown regular expression.*

*Proof sketch.* We first prove that ii $\implies$ i. The proof is by structural induction on a visibly pushdown regular expression. The base cases 0, 1, $a \in \Sigma_{\mathbf{i}}$, $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$ are easy:




The induction cases for $\cdot$, $+$ and $^*$ are immediate from [2, Theorems 3.5 and 3.6]. So, the case for the set of grammar patterns $\mathcal{G}$ is the only one remaining. Of course, a grammar pattern along with its operands is a WMVPG. By Theorem 3.8, there exists a VPG that generates the same language as this WMVPG. Thus, we just have to prove that there exists a visibly pushdown automaton that accepts the same language as the one generated by this VPG. This is immediate from [2, Theorem 5.3].

We now show that i $\implies$ ii. To do this, we use matrices over the languages contained in $2^{(\Sigma_{\mathbf{i}} \cup \Sigma_{\mathbf{c}} \cup \Sigma_{\mathbf{r}})^*}$. In other words, we use matrices whose entries are languages. Of course, we can use visibly pushdown regular expressions to denote languages in the entries of these matrices. So, we can use the matrix operations $\bullet$, $+$ and $^\star$ defined in Section 2.1 between matrices on VPREs. These matrices will ease the calculation of the visibly pushdown regular expression representing a visibly pushdown automaton.

Let $S := \{s_1, s_2, \ldots, s_{|S|}\}$ be a nonempty finite set. Encode a visibly pushdown automaton

$$(S, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma \cup \{\bot\}, \delta, I, F)$$

by the structure

$$(\overrightarrow{I}, \mathbf{WM}, \mathbf{T}_{\mathbf{c}}, \mathbf{T}_{\bot}, \overrightarrow{F})$$

where:

- $\overrightarrow{I}$ and $\overrightarrow{F}$ are "Boolean" column vectors[2] of size $|S|$ representing the initial states and the accepting states;

[2]Boolean column vectors in this case are vectors that contain only 0s and 1s where 0 and 1 are the constants of VPREs.

- each entry $i, j$ of the square matrix $\mathbf{T_c}$ (of size $|S| \times |S|$) is defined by

$$\mathbf{T_c}[i,j] := (\sum a \mid a \in \Sigma_{\mathbf{c}} \wedge (\exists\, d \mid d \in \Gamma : (s_i, a, \lambda; s_j, d) \in \delta) : a) \ .$$

  This matrix represents the possibly pending calls;

- each entry $i, j$ of the square matrix $\mathbf{T}_\perp$ (of size $|S| \times |S|$) is defined by

$$\mathbf{T}_\perp[i,j] := (\sum a \mid a \in \Sigma_{\mathbf{r}} \wedge (s_i, a, \perp; s_j, \perp) \in \delta : a) \ .$$

  This matrix represents the possibly pending returns;

- each entry $i, j$ of the square matrix $\mathbf{WM}$ (of size $|S| \times |S|$) represents the well-matched words that can be generated from a state $s_i$ of the automaton to a state $s_j$, and it is defined by

$$\mathbf{WM}[i,j] := \underset{s_i}{\overset{s_j}{(\!|\, \mathcal{B} \,|\!)}}$$

  where the list of blocks $\mathcal{B}$ (identical for all entries of the matrix) represents the encoding of the structure of the visibly pushdown automaton using the following rules:

  A. The set of labels $S$ is used.

  B. For all states $s_i \in S$, $\underset{s_i}{[\,} 1 \,]^{s_i} \in \mathcal{B}$.

  C. For all internal actions $a \in \Sigma_{\mathbf{i}}$ and states $s_i, s_j \in S$,

$$\underset{s_i}{[\,} a \,]^{s_j} \in \mathcal{B} \Leftrightarrow (s_i, a, \lambda; s_j, \lambda) \in \delta \ .$$

  D. For all call actions $c \in \Sigma_{\mathbf{c}}$, return actions $r \in \Sigma_{\mathbf{r}}$ and states $s_i, s_j, s_k, s_l \in S$,

$$\underset{s_i}{[\,} c \underset{s_k}{\downarrow} \overset{s_j}{\uparrow} r \,]^{s_l} \in \mathcal{B} \Leftrightarrow (\exists\, d \mid d \in \Gamma : (s_i, c, \lambda; s_k, d) \in \delta \wedge (s_j, r, d; s_l, \lambda) \in \ \delta) \ .$$

The encoding is mainly concerned about getting rid of the explicit stack in the automaton. Note that the exact stack symbol pushed on the stack when reading a call action is important only if this symbol is used to read a return action. So, this situation occurs only if an affix of the current word is well-matched and contains these call and return actions. So, this situation can be taken care of without using an explicit stack by using the matrix $\mathbf{WM}$ which allows a binary block only if the call action and the return action use the same stack symbol. In all other cases, it is not useful to know the exact stack symbol used, but it is essential to know whether the stack is empty of not. So, the encoding forgets the stack symbol for the matrix $\mathbf{T_c}$.

Using this encoding, we find the visibly pushdown regular expression that represents exactly the language accepted by the visibly pushdown automaton. It is the expression contained in the entry of the $1 \times 1$ matrix:

$$\overrightarrow{I}^{t} \bullet (\mathbf{T}_{\perp} + \mathbf{WM})^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet \overrightarrow{F} \quad . \tag{3.1}$$

This is inspired from Equation (2.2). Intuitively, assume that $\mathbf{WM}[i, j]$ gives the set of well-matched words that can be read by the VPA. This assumption is easy to accept because the expression encodes the structure of the automaton. So, the language (3.1) can be understood in the following way:

- First, any accepted word of the automaton starts in an initial state (a state of $\overrightarrow{I}$) with an empty stack.
- The expression $(\mathbf{T}_{\perp} + \mathbf{WM})^{\star}$ represents the possibilities that the automaton has to accept the word *without leaving an empty stack*:
  - it can read a pending return action (transition) of $\mathbf{T}_{\perp}$ (so, the stack remains the same);
  - it can read a well-matched word by using $\mathbf{WM}$ (so, the stack remains the same).
- The expression $(\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star}$ represents the possibilities that the automaton has to accept a word when processing the first pending call: reading a pending call action followed by well-matched words and more pending calls.
- The automaton must stop in a final state (a state of $\overrightarrow{F}$). ■

### 3.1.2 Block-Based Definition of Visibly Pushdown Regular Expressions

An alternative definition of the infinite family of operators is to define it directly with the notion of "block" instead of using the detour of grammars. Thus, in this definition, visibly pushdown regular expressions are an extension of regular expressions with an infinite family of operators working on finite lists of "blocks".

There are two kinds of blocks:

**Unary blocks** of the form $[_x\, m\, ]^y$, where $m \in \Sigma_{\mathbf{i}} \cup \{\varepsilon\}$. The labels $x$ and $y$ are respectively called the *starting label* and the *ending label*. The element $m$ is called the *operand* of the unary block;

**Binary blocks** of the form $[_x\, c{\downarrow_z}{\uparrow^w}\, r\,]^y$, where $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$. The labels $x$, $y$, $z$ and $w$ are respectively called the *starting label*, the *ending label*, the *call label* and the *return label*. The elements $c$ and $r$ are respectively called the *left operand* and the *right operand* of the binary block.

Let $\mathcal{B}$ be a *finite* list of unary and binary blocks on a *finite* set of labels $V$, and let $x, x' \in V$. An operator of the family has the form $(\!|_x\, \mathcal{B}\, |\!)^{x'}$. The operator's arity is the number of unary blocks in $\mathcal{B}$ that do not contain $\varepsilon$ as operand plus twice the number of binary blocks in $\mathcal{B}$. Also, note that the labels in an operator are not variables but just a way to identify which operator of the family is used. To see this more clearly, let $\Sigma_{\mathbf{i}} := \{\mathsf{a}, \mathsf{b}\}$, $\Sigma_{\mathbf{c}} := \{\mathsf{c}, \mathsf{d}\}$, $\Sigma_{\mathbf{r}} := \{\mathsf{r}, \mathsf{s}\}$ and $V := \{\mathsf{v}, \mathsf{w}, \mathsf{x}, \mathsf{y}, \mathsf{z}\}$, and rewrite the expression

$$(\!|\, [_{\mathsf{x}}\, \mathsf{c}{\downarrow_{\mathsf{w}}}{\uparrow^{\mathsf{v}}}\, \mathsf{r}\,]^{\mathsf{z}},\ [_{\mathsf{y}}\, \mathsf{b}\,]^{\mathsf{z}},\ [_{\mathsf{w}}\, \mathsf{d}{\downarrow_{\mathsf{x}}}{\uparrow^{\mathsf{z}}}\, \mathsf{s}\,]^{\mathsf{v}},\ [_{\mathsf{x}}\, \mathsf{a}\,]^{\mathsf{y}\mathsf{z}}\, |\!)$$

by the expression

$$f_{(\mathsf{x},(\mathsf{y},\mathsf{w},\mathsf{v},\mathsf{z}),(\mathsf{y},\mathsf{z}),(\mathsf{w},\mathsf{x},\mathsf{z},\mathsf{v}),(\mathsf{x},\mathsf{y}),\mathsf{z})}(\mathsf{c}, \mathsf{r}, \mathsf{b}, \mathsf{d}, \mathsf{s}, \mathsf{a})$$

in which $f_{(\mathsf{x},(\mathsf{y},\mathsf{w},\mathsf{v},\mathsf{z}),(\mathsf{y},\mathsf{z}),(\mathsf{w},\mathsf{x},\mathsf{z},\mathsf{v}),(\mathsf{x},\mathsf{y}),\mathsf{z})}$ is a 6-ary operator. Moreover, note that each operator of the family is partial. An operand of a block is just an element allowed by the definition of blocks.

The idea behind an expression $(\!|_x\, \mathcal{B}\, |\!)^{x'}$ is to generate any well-matched word that can be produced by a correct "travelling" of the list of blocks, starting the travel in any block that has $x$ as starting label and ending it in any block that has $x'$ as ending label. A correct travelling starting with $y$, ending with $y'$ and producing a set of well-matched words $S$ is a finite sequence $b_1 b_2 \dots b_n$ of blocks of $\mathcal{B}$ (where $n > 0$), such that either $n = 1$ and $b_n$ is a unary block of the form $[_y\, m\,]^{y'}$ and $S = \{m\}$, or $n > 1$ and (there are three possible cases):

- $b_1$ is a unary block of the form $[_y\, m\,]^{v'}$ for $v' \in V$ (including $y'$) and $b_2 \dots b_n$ is a correct travelling starting with $v'$, ending with $y'$ and producing $T$, and $S = \{m\} \bullet T$;
- $b_1$ is a binary block of the form $[_y\, c{\downarrow_z}{\uparrow^w}\, r\,]^{y'}$ and $b_2 \dots b_n$ is a correct travelling starting with $z$, ending with $w$ and producing $T$, and $S = \{c\} \bullet T \bullet \{r\}$;
- $b_1$ is a binary block of the form $[_y\, c{\downarrow_z}{\uparrow^w}\, r\,]^{v'}$ for $v' \in V$ (including $y'$) and there exists an $i \in \mathbb{N}$ such that $1 < i < n$ and $b_2 \dots b_i$ is a correct travelling starting with $z$, ending with $w$ and producing $T$ and $b_{i+1} \dots b_n$ is a correct travelling starting with $v'$, ending with $y'$ and producing $U$, and $S = \{c\} \bullet T \bullet \{r\} \bullet U$.

Note that the idea of a correct travelling of a $(\!|\,|\!)$-expression is similar to the derivation relation in grammars restricted to well-matched VPLs using a forward strategy.

Let $\mathcal{B}$ be a finite list of unary and binary blocks on a finite set of labels $V$. Define $\mathcal{B}^1$ as the set of unary blocks of $\mathcal{B}$ and $\mathcal{B}^2$ as the set of binary blocks of $\mathcal{B}$. For $n \in \mathbb{N}$, define the *power-recursion operator* $(\!|_x\, \mathcal{B}\, |\!)^y_n$, where $x, y \in V$, by induction on $n$: $(\!|_x\, \mathcal{B}\, |\!)^y_0 := (\cup\, m \mid [_x\, m\, ]^y \in \mathcal{B}^1 : \{m\})$, and

$$
\begin{aligned}
(\!|\, \mathcal{B}\, |\!)^{\,y}_{x\quad n+1} := {} & (\cup\, m, v \mid [\,{}^{\,v}_{x}\, m\,] \in \mathcal{B}^1 : \{m\} \bullet (\!|\, \mathcal{B}\, |\!)^{\,y}_{v\quad n}) \\
& \cup\ (\cup\, c, z, r, w \mid [\,{}^{\,w\quad y}_{x\quad z}\, c{\downarrow}{\uparrow} r\,] \in \mathcal{B}^2 : \{c\} \bullet (\!|\, \mathcal{B}\, |\!)^{\,w}_{z\quad n} \bullet \{r\}) \\
& \cup\ (\cup\, c, z, r, w, v, n_1, n_2 \mid [\,{}^{\,w\quad v}_{x\quad z}\, c{\downarrow}{\uparrow} r\,] \in \mathcal{B}^2 \wedge\ n_1, n_2 \in \mathbb{N} \\
& \qquad \wedge\ n_1 + n_2 = n - 1 : \{c\} \bullet (\!|\, \mathcal{B}\, |\!)^{\,w}_{z\quad n_1} \bullet \{r\} \bullet (\!|\, \mathcal{B}\, |\!)^{\,y}_{v\quad n_2})\ .
\end{aligned}
$$

Intuitively, $(\!|_x\, \mathcal{B}\, |\!)^y_n$ denotes the set of all well-matched words that can be generated by any correct travelling of $\mathcal{B}$ of length $n+1$ starting with $x$ and ending with $y$. With this definition, it is easy to define an operator $(\!|_x\, \mathcal{B}\, |\!)^y$ by

$$
(\!|\, \mathcal{B}\, |\!)^{\,y}_{x} := (\cup\, n \mid n \in \mathbb{N} : (\!|\, \mathcal{B}\, |\!)^{\,y}_{x\quad n})\ .
$$

We are now ready to define visibly pushdown regular expressions.

**Definition 3.10** (Visibly pushdown regular expressions (block-based))**.** Let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be disjoint finite sets of atomic elements. Define a *labelled block pattern* of a set expression $(\!|_x\, \mathcal{B}\, |\!)^{x'}$ on $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ to be a partial operator obtained by

- replacing the operands (except $\varepsilon$) in blocks by variables (placeholders) of the same type ($\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ or $\Sigma_{\mathbf{r}}$);
- writing $[_x\, 1\, ]^y$ instead of $[_x\, \varepsilon\, ]^y$;
- adding, for convenience, blocks of the form $[_x\, 0\, ]^y$.

The arity of a labelled block pattern is the arity of the set expression $(\!|_x\, \mathcal{B}\, |\!)^{x'}$ that it represents. Define $\mathcal{F}_{(\!|\,|\!)}$ as the set of all labelled block patterns on $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$. A VPRE is any well-formed expression that can be generated from the base elements 0, 1, $a$ for each $a \in \Sigma_{\mathbf{i}} \cup \Sigma_{\mathbf{c}} \cup \Sigma_{\mathbf{r}}$, the unary operator $^*$, the binary operators $\cdot$ and $+$, and

the operators in $\mathcal{F}_{(\!|\,|\!)}$. The language denoted by a VPRE $p$ is noted $\mathcal{L}(p)$ and is defined by

$$\mathcal{L}(0) := \emptyset, \qquad \mathcal{L}(1) := \{\varepsilon\}, \qquad \mathcal{L}(a) := \{a\} \text{ for any } a \in \Sigma_{\mathbf{i}} \cup \Sigma_{\mathbf{c}} \cup \Sigma_{\mathbf{r}},$$

and extends over the structure of VPREs with $\cdot$ becoming $\bullet$, $+$ becoming $\cup$, and $^*$ becoming the set operator $^*$. An operator of $\mathcal{F}_{(\!|\,|\!)}$ along with its operands becomes a set expression $(\!|_x\, \mathcal{B}\,|\!)^{x'}$, and so $\mathcal{L}((\!|_x\, \mathcal{B}\,|\!)^{x'})$ is the language denoted by $(\!|_x\, \mathcal{B}\,|\!)^{x'}$.[3] ♦

This definition of VPRE is equivalent to the definition given in Section 3.1.1 [11].

**Theorem 3.11.** *The definitions of* VPRE *given in Section 3.1.1 and in Section 3.1.2 represent the same class of languages.*

*Proof.* By Theorem 3.4, it suffices to prove that a $(\!|\,|\!)$-expression written with an operator of $\mathcal{F}_{(\!|\,|\!)}$ gives exactly the same language as if it was written with its associated grammar restricted to well-matched VPLs using a forward strategy and vice versa.

Let $\mathcal{B}$ be a finite list of unary and binary blocks on a finite set of labels $V$. It suffices to prove that, for all $n \in \mathbb{N}$ and $x, y \in V$, the language represented by the expression $(\!|_x\, \mathcal{B}\,|\!)^y_n$ is exactly the set of well-matched words that can be generated by the associated grammar restricted to well-matched VPLs using a forward strategy $(\!|_x\, \mathcal{B}\,|\!)^y$ when using exactly $n+1$ rewrite rules and vice versa. We do a proof by generalized induction over $n$.

For the base case $n = 0$, $(\!|_x\, \mathcal{B}\,|\!)^y_0 = (\cup\, m \mid [_x\, m\,]^y \in \mathcal{B}^1 : \{m\})$. So, every word of $(\!|_x\, \mathcal{B}\,|\!)^y_0$ is an operand of a unary block that begins with $x$ and ends with $y$. This is the same situation for the language generated by the associated grammar restricted to well-matched VPLs using a forward strategy $(\!|_x\, \mathcal{B}\,|\!)^y$ when using exactly 1 rewrite rule.

For the inductive case, we suppose that for all $j \in \{0, 1, \ldots, k\}$ and $x, y \in V$, the language represented by the expression $(\!|_x\, \mathcal{B}\,|\!)^y_j$ is exactly the set of well-matched words that can be generated by the associated grammar restricted to well-matched VPLs using a forward strategy $(\!|_x\, \mathcal{B}\,|\!)^y$ when using exactly $j+1$ rewrite rules and vice versa. We prove that for all $x, y \in V$, the language represented by the expression $(\!|_x\, \mathcal{B}\,|\!)^y_{k+1}$ is exactly the set of well-matched words that can be generated by the associated grammar restricted to well-matched VPLs using a forward strategy $(\!|_x\, \mathcal{B}\,|\!)^y$ when using exactly $k+2$ rewrite

[3] Blocks of the form $[_x\, 0\,]^y$ do not translate to a block. They are just omitted in the set operator. Again, those blocks are a way to identify the `fail` instruction in programs.

rules and vice versa. By the definition of the family $\mathcal{F}_{\langle\rangle}$,

$$\begin{aligned}
\langle\!\!\langle \mathcal{B} \rangle\!\!\rangle_{x\ k+1}^{\ y} := (\cup\, m, v \mid [\,_{x}\, m\,]^{v} \in \mathcal{B}^1 : \{m\} \bullet \langle\!\!\langle \mathcal{B} \rangle\!\!\rangle_{v\ k}^{\ y}) \\
\cup\ (\cup\, c, z, r, w \mid [\,_{x}\, c \downarrow_{z}^{w} \uparrow r\,]^{y} \in \mathcal{B}^2 : \{c\} \bullet \langle\!\!\langle \mathcal{B} \rangle\!\!\rangle_{z\ k}^{\ w} \bullet \{r\}) \\
\cup\ (\cup\, c, z, r, w, v, k_1, k_2 \mid [\{c\} \downarrow_{z}^{w} \uparrow \{r\}\,]^{v} \in \mathcal{B}^2 \wedge\ k_1, k_2 \in \mathbb{N} \\
\wedge\ k_1 + k_2 = k - 1 : \{c\} \bullet \langle\!\!\langle \mathcal{B} \rangle\!\!\rangle_{z\ k_1}^{\ w} \bullet \{r\} \bullet \langle\!\!\langle \mathcal{B} \rangle\!\!\rangle_{v\ k_2}^{\ y})\ .
\end{aligned}$$

Every word $w$ of the first union quantifier can be derived by the associated grammar restricted to well-matched VPLs using a forward strategy by using exactly $k+2$ rewrite rules. First, the induction hypothesis states that the language represented by the expression $\langle\!\!\langle_{v} \mathcal{B} \rangle\!\!\rangle_{k}^{y}$ is exactly the set of well-matched words that can be generated by the associated grammar restricted to well-matched VPLs using a forward strategy $\langle\!\!\langle_{v} \mathcal{B} \rangle\!\!\rangle^{y}$ when using exactly $k+1$ rewrite rules and vice versa. Then, it suffices to use the implicit rewrite rule $P_{(x,y)} \to m\, P_{(v,y)}$ (where $m \in \Sigma_{\mathbf{i}}$ or $m = \varepsilon$) to generate $w$. The opposite direction is also true.

Every word $w$ of the second union quantifier can be derived by the associated grammar restricted to well-matched VPLs using a forward strategy by using exactly $k+2$ rewrite rules. First, the induction hypothesis states that the language represented by the expression $\langle\!\!\langle_{z} \mathcal{B} \rangle\!\!\rangle_{k}^{w}$ is exactly the set of well-matched words that can be generated by the associated grammar restricted to well-matched VPLs using a forward strategy $\langle\!\!\langle_{z} \mathcal{B} \rangle\!\!\rangle^{w}$ when using exactly $k+1$ rewrite rules and vice versa. Then, it suffices to use the explicit rewrite rule $P_{(x,y)} \to c\, P_{(z,w)}\, r$ to generate $w$. The opposite direction is also true.

Every word $w$ of the third union quantifier can be derived by the associated grammar restricted to well-matched VPLs using a forward strategy by using exactly $k+2$ rewrite rules. First, the induction hypothesis states that the languages represented by the expressions $\langle\!\!\langle_{z} \mathcal{B} \rangle\!\!\rangle_{k_1}^{w}$ and $\langle\!\!\langle_{v} \mathcal{B} \rangle\!\!\rangle_{k_2}^{y}$ are exactly respectively the set of well-matched words that can be generated by the associated grammar restricted to well-matched VPLs using a forward strategy $\langle\!\!\langle_{z} \mathcal{B} \rangle\!\!\rangle^{w}$ when using exactly $k_1+1$ rewrite rules and $\langle\!\!\langle_{v} \mathcal{B} \rangle\!\!\rangle^{y}$ when using exactly $k_2+1$ rewrite rules and vice versa. So, $k_1 + 1 + k_2 + 1 = k - 1 + 2 = k + 1$ rewrite rules have currently been used. Then, it suffices to use the implicit rewrite rule $P_{(x,y)} \to c\, P_{(z,w)}\, r\, P_{(v,y)}$ to generate $w$. The opposite direction is also true. ■

We will favour the grammar-based definition in this dissertation particularly when explaining intuition behind laws and reasonings.

## 3.2 Axiomatic System of Visibly Pushdown Kleene Algebra

In this section, we first present the axiomatization of visibly pushdown Kleene algebra (VPKA). Then, we show some useful laws of VPKA in Section 3.2.2 and we also compare the axiomatic system of VPKA with other algebras that axiomatize subclasses of context-free languages in Section 3.2.3.

### 3.2.1 Axiomatization of Visibly Pushdown Kleene Algebra

Visibly Pushdown Kleene Algebra characterizes exactly the equality of the languages denoted by two VPREs. Its axiomatization adds seven axioms to Kleene algebra to deal with grammar patterns. The first two axioms of VPKA are inspired by the *explicit* rewrite rules of a WMVPG. The first one states that a WMVPG $(V, P_{(x,y)}, \to)$ can generate at least all terminals of explicit rewrite rules whose left-hand side is the starting nonterminal $P_{(x,y)}$ and right-hand side a single terminal or the empty word. Thus, let $(\!|_x \mathcal{B} |\!)^y$ be an expression, on a finite set of symbols $V$, representing an operator of $\mathcal{G}$ along with its operands. The first axiom of VPKA is

$$\underset{x}{(\!|} \mathcal{B} \overset{y}{|\!)} \geqslant m, \qquad \text{for } \underset{x}{[} m \overset{y}{]} \in \mathcal{B}^1 \ . \tag{3.2}$$

Note that, in the algebraic world, the "generation of a word" is replaced by the operator "greater or equals" ($\geqslant$) between expressions since it operates like the $\supseteq$ operator of sets. Moreover, note that axiom (3.2) also states that if $[_x\, 0\,]^y \in \mathcal{B}^1$, then $0 \leqslant (\!|_x \mathcal{B} |\!)^y$. This adds nothing to the axiomatization since, by Kleene algebra, 0 is the least element of the algebra. After all, the empty set is a subset of any set.

The second axiom of VPKA represents the explicit rewrite rules of a WMVPG whose right-hand side is complex (it contains a call terminal, a return terminal, and a nonterminal between the terminals). It is presented similarly as axiom (3.2). Thus, let $(\!|_x \mathcal{B} |\!)^y$ be an expression, on a finite set of symbols $V$, representing an operator of $\mathcal{G}$ along with its operands and let $z, w \in V$. The second axiom of VPKA is

$$\underset{x}{(\!|} \mathcal{B} \overset{y}{|\!)} \geqslant c \cdot \underset{z}{(\!|} \mathcal{B} \overset{w}{|\!)} \cdot r, \qquad \text{for } \underset{x}{[}\, c \underset{z}{\downarrow} \overset{w}{\uparrow} r \,\overset{y}{]} \in \mathcal{B}^2 \ . \tag{3.3}$$

Note that axiom (3.3) uses the expression $(\!|_z \mathcal{B} |\!)^w$ to represent the nonterminal $P_{(z,w)}$. This is easily understood when thinking of grammars. Suppose that a WMVPG $G := (V, P_{(x,y)}, \to)$ has an explicit rewrite rule of the form $P_{(x,y)} \to c\, P_{(z,w)}\, r$. Thus, the

derivation relation can be used to have $P_{(x,y)} \Rightarrow c\, P_{(z,w)}\, r$. Since $c$ and $r$ are terminals, only $P_{(z,w)}$ can be derived further. So, at this point, if we want to know $c\, P_{(z,w)}\, r \Rightarrow^* ?$ (i.e., which words can be derived by $c\, P_{(z,w)}\, r$?), then the problem is to derive all words that can be generated by $G$ when starting with the nonterminal $P_{(z,w)}$ and concatenating $\{c\}$ and $\{r\}$ respectively before and after the language generated. So, the derivation part of the problem is exactly the same as using a grammar $G'$ that is identical to $G$ except that the starting nonterminal is now $P_{(z,w)}$. This is what axiom (3.3) does.

The third axiom of VPKA represents the *implicit* rewrite rules of the grammar. Thus, let $(\!|_x\, \mathcal{B}\, |\!)^y$ be an expression, on a finite set of symbols $V$, representing an operator of $\mathcal{G}$ along with its operands and let $z \in V$. The third axiom of VPKA is

$$(\!|\, \mathcal{B}\, |\!)^{y}_{x} \geqslant (\!|\, \mathcal{B}\, |\!)^{z}_{x} \cdot (\!|\, \mathcal{B}\, |\!)^{y}_{z} \quad . \tag{3.4}$$

Once again, we use expressions like $(\!|_x\, \mathcal{B}\, |\!)^z$ to represent a nonterminal $P_{(x,z)}$. The idea behind that is similar to the idea used in axiom (3.3). Suppose that a WMVPG $G := (V, P_{(x,y)}, \rightarrow)$ has an implicit rewrite rule of the form $P_{(x,y)} \rightarrow P_{(x,z)}\, P_{(z,y)}$. Thus, the derivation relation can be used to have $P_{(x,y)} \Rightarrow P_{(x,z)}\, P_{(z,y)}$. Both $P_{(x,z)}$ and $P_{(z,y)}$ can be derived further. So, at this point, if we want to know $P_{(x,z)}\, P_{(z,y)} \Rightarrow^* ?$ (i.e., which words can be derived by $P_{(x,z)}\, P_{(z,y)}$?), then the problem can be split in two parts:

1. derive all words that can be generated by $G$ when starting with the nonterminal $P_{(x,z)}$;
2. derive all words that can be generated by $G$ when starting with the nonterminal $P_{(z,y)}$;

and the overall problem is resolved by concatenating the two languages generated. So, the first derivation part of the problem is exactly the same as using a grammar $G'$ that is identical to $G$ except that the starting nonterminal is now $P_{(x,z)}$, and the second derivation part of the problem is exactly the same as using a grammar $G''$ that is identical to $G$ except that the starting nonterminal is now $P_{(z,y)}$. This is what axiom (3.4) does.

Here, we can stop and think of the significance of axioms (3.2) to (3.4). These axioms are a convenient way to simulate the derivation of *every* word of a WMVPG. This claim might surprise the reader. After all, axioms (3.2) and (3.3) only deal with explicit rewrite rules whose left-hand side is the *starting* nonterminal. They do not deal with *every* left-hand side! However, axiom (3.4), applied wisely, helps us to handle every

left-hand side $P_{(z,w)}$ of an explicit rewrite rule of an expression $(\!|_x \mathcal{B} |\!)^y$. For example, if $z$ and $w$ are not $x$ nor $y$, then we can use axiom (3.4) twice to obtain

$$\underset{x}{(\!|} \mathcal{B} \overset{y}{|\!)} \geqslant \underset{x}{(\!|} \mathcal{B} \overset{z}{|\!)} \cdot \underset{z}{(\!|} \mathcal{B} \overset{w}{|\!)} \cdot \underset{w}{(\!|} \mathcal{B} \overset{y}{|\!)} \quad .$$

So, it is always possible to simulate the derivation of *every* word of a WMVPG by using axioms (3.2) to (3.4).

Being able to prove that a word is really generated by a given grammar does not handle all the cases of equality of the languages denoted by two VPREs. Of course, axioms (3.2) to (3.4) are able to prove equalities other than those related to the derivation of word, but they cannot handle all equalities between languages denoted by two VPREs. In particular, there is no axiom stating if the language generated by a grammar is included into (is a subset of) another language. The next two axioms handle most of these cases. To explain them, we take a detour via inequational systems described by a grammar. A grammar can be described by an inequational system by considering that every nonterminal is a variable and by converting its rewrite rules $P \to \alpha$ to an inequation $P \geqslant \alpha$. Of course, it is possible to write an expression $\alpha$ containing terminals and variables $P_i$ as $\alpha(P_1, P_2, \ldots, P_n)$ to emphasize possible use of the variables in the expression $\alpha$. Thus, a grammar is translated to an inequational system

$$\begin{aligned}
P_1 &\geqslant \alpha_1(P_1, P_2, \ldots, P_n) \\
P_2 &\geqslant \alpha_2(P_1, P_2, \ldots, P_n) \\
P_2 &\geqslant \alpha_3(P_1, P_2, \ldots, P_n) \\
&\vdots \\
P_n &\geqslant \alpha_m(P_1, P_2, \ldots, P_n)
\end{aligned}$$

Note that the third line in the inequational system above begins with $P_2$. This is to make explicit that each rewrite rule is written as an inequation, so if a nonterminal (now, a variable) is the left-hand side of more than one rewrite rule, then it will appear more than once in the inequational system.

It is natural to use a complete *solution* vector for an inequation system. A vector $\overrightarrow{s}$ is a solution of an inequational system if each inequation of the system is valid when substituting all variables $P_i$ by their component solution $\overrightarrow{s}(i)$. So, $\overrightarrow{s}$ contains a component solution for all variables of the inequational system, and only that. When viewed as a solution of an inequational system, the language generated by a grammar is the *least solution* for the component (variable) representing the starting nonterminal.

We will use the analogy with inequational systems to define the next two induction axioms called $(\!|\,|\!)$*-induction axioms*. They define an expression $(\!|_x \mathcal{B} |\!)^y$ as the least solution for the nonterminal $P_{(x,y)}$ of the inequational system described by the grammar.

So, they state that $(\!|_x \mathcal{B} |\!)^y$ is less than or equal to any other possible solution $\overrightarrow{s}$ of the inequational system for the component $P_{(x,y)}$. The hypothesis part of the induction only validates that $\overrightarrow{s}$ is a solution of the inequational system. Of course, the validation proceeds by verifying if each inequation of the system is valid when substituting all variables $P_{(u,u')}$ for $u, u' \in V$ by their component solution $\overrightarrow{s}(u, u')$.

From the definition of WMVPGs, it seems natural to define an axiom such as the following. Let $(\!|_x \mathcal{B} |\!)^y$ be an expression on a finite set of symbols $V$ representing an operator of $\mathcal{G}$ along with its operands. Let $s_{(u,u')}$ be a VPRE for all $u, u' \in V$. Each $s_{(u,u')}$ acts as a solution for a component (or nonterminal) $P_{(u,u')}$ of the grammar. Collectively, they represent a complete solution vector $\overrightarrow{s}$. Then

$$\begin{array}{l}\Big(\wedge\, u, u' \mid u, u' \in V : \\ \qquad (\wedge\, m \mid [\underset{u}{m}\,]^{u'} \in \mathcal{B}^1 : m \leqslant s_{(u,u')}) \\ \quad \wedge\ (\wedge\, c, z, r, w \mid [\underset{u}{c}\underset{z}{\downarrow}\overset{w}{\uparrow} r\,]^{u'} \in \mathcal{B}^2 : c \cdot s_{(z,w)} \cdot r \leqslant s_{(u,u')}) \\ \quad \wedge\ (\wedge\, v \mid v \in V : s_{(u,v)} \cdot s_{(v,u')} \leqslant s_{(u,u')})\Big) \\ \rightarrow\ (\!|\underset{x}{\mathcal{B}}|\!)^y \leqslant s_{(x,y)}\ .\end{array} \tag{3.5}$$

Note that the hypotheses of (3.5) represent the validation of the solution $\overrightarrow{s}$ for the inequational system of a WMVPG. So, it contains inequations representing explicit rewrite rules and implicit rewrite rules in which the nonterminals have been substituted by their component solution. Thus, if the hypotheses of (3.5) are valid for given $s_{(u,u')}$ for all $u, u' \in V$, then the law states that $(\!|_x \mathcal{B} |\!)^y$ is smaller than or equal to the component solution $s_{(x,y)}$. Note also that $s_{(u,u')} := (\!|_u \mathcal{B} |\!)^{u'}$ for all $u, u' \in V$ is a solution of the inequational system (simply use axioms (3.2) to (3.4)). So, the law (3.5) states that $(\!|_x \mathcal{B} |\!)^y$ is the least solution for the nonterminal $P_{(x,y)}$ of the inequational system described by the grammar.

However, it is not convenient to take (3.5) as an axiom. The problem is that the verification of the last set of hypotheses

$$(\wedge\, u, u' \mid u, u' \in V : (\wedge\, v \mid v \in V : s_{(u,v)} \cdot s_{(v,u')} \leqslant s_{(u,u')}))$$

can be difficult to do. There can be lots of cases to verify and some of them may be very complex. Thus, it is convenient to take as axioms other laws that contain simpler conditions for the same goal. Recall from Section 3.1.1 that WMVPGs are equivalent to grammars restricted to well-matched VPLs using a forward strategy. So, this definition of grammar can be taken and an inequational system can be extracted from it (as was done with WMVPGs). Let $(\!|_x \mathcal{B} |\!)^y$ be an expression on a finite set of symbols $V$

representing an operator of $\mathcal{G}$ along with its operands. Let $s_{(u,u')}$ be a VPRE for all $u, u' \in V$. Each $s_{(u,u')}$ acts as a solution for a component (or nonterminal) $P_{(u,u')}$ of the grammar. Collectively, they represent a complete solution vector $\overrightarrow{s}$. Then

$$\boxed{\begin{array}{l}\Big(\wedge\, u, u' \mid u, u' \in V : \\ \qquad (\wedge\, m \mid [\,{}_{u}\, m\,{}^{u'}\,] \in \mathcal{B}^1 : m \leqslant s_{(u,u')}) \\ \quad \wedge\ (\wedge\, m, v \mid [\,{}_{u}\, m\,{}^{v}\,] \in \mathcal{B}^1 : m \cdot s_{(v,u')} \leqslant s_{(u,u')}) \\ \quad \wedge\ (\wedge\, c, z, r, w \mid [\,{}_{u}\, c\downarrow{}_{z}^{w}\uparrow r\,{}^{u'}\,] \in \mathcal{B}^2 : c \cdot s_{(z,w)} \cdot r \leqslant s_{(u,u')}) \\ \quad \wedge\ (\wedge\, c, z, r, w, v \mid [\,{}_{u}\, c\downarrow{}_{z}^{w}\uparrow r\,{}^{v}\,] \in \mathcal{B}^2 : c \cdot s_{(z,w)} \cdot r \cdot s_{(v,u')} \leqslant s_{(u,u')})\Big) \\ \to\ (\!|\,{}_{x}\,\mathcal{B}\,{}^{y}\,|\!) \leqslant s_{(x,y)}\ .\end{array}} \tag{3.6}$$

The hypotheses in this law should be simpler to verify most of the time because each one is linked to the existence of a rewrite rule in the grammar and the body of each quantification contains more precise information on the left-hand side of $\leqslant$. Note that there could be more hypotheses to prove in (3.6) for a given formula than in (3.5). However, law (3.6) and Kleene algebra allow one to derive law (3.5) as shown below.

To prove (3.5) from (3.6), first suppose, for all $u, u' \in V$,

$$(\wedge\, m \mid [\,{}_{u}\, m\,{}^{u'}\,] \in \mathcal{B}^1 : m \leqslant s_{(u,u')})\ , \tag{3.7}$$

$$(\wedge\, c, z, r, w \mid [\,{}_{u}\, c\downarrow{}_{z}^{w}\uparrow r\,{}^{u'}\,] \in \mathcal{B}^2 : c \cdot s_{(z,w)} \cdot r \leqslant s_{(u,u')})\ , \tag{3.8}$$

$$(\wedge\, v \mid v \in V : s_{(u,v)} \cdot s_{(v,u')} \leqslant s_{(u,u')})\ . \tag{3.9}$$

We prove

$$(\!|\,{}_{x}\,\mathcal{B}\,{}^{y}\,|\!) \leqslant s_{(x,y)}\ .$$

By (3.6) with $s_{(u,u')} := s_{(u,u')}$ for all $u, u' \in V$, it suffices to prove that, for all $u, u' \in V$

$$(\wedge\, m \mid [\,{}_{u}\, m\,{}^{u'}\,] \in \mathcal{B}^1 : m \leqslant s_{(u,u')})\ , \tag{3.10}$$

$$(\wedge\, m, v \mid [\,{}_{u}\, m\,{}^{v}\,] \in \mathcal{B}^1 : m \cdot s_{(v,u')} \leqslant s_{(u,u')})\ , \tag{3.11}$$

$$(\wedge\, c, z, r, w \mid [\,{}_{u}\, c\downarrow{}_{z}^{w}\uparrow r\,{}^{u'}\,] \in \mathcal{B}^2 : c \cdot s_{(z,w)} \cdot r \leqslant s_{(u,u')})\ , \tag{3.12}$$

$$(\wedge\, c, z, r, w, v \mid [\,{}_{u}\, c\downarrow{}_{z}^{w}\uparrow r\,{}^{v}\,] \in \mathcal{B}^2 : c \cdot s_{(z,w)} \cdot r \cdot s_{(v,u')} \leqslant s_{(u,u')})\ . \tag{3.13}$$

Inequations (3.10) and (3.12) are exactly hypotheses (3.7) and (3.8). We now prove (3.11). For each unary block $[_u\, m\,]^v \in \mathcal{B}^1$, we prove that

$$m \cdot s_{(v,u')} \leqslant s_{(u,u')} \ .$$

We are able to prove it.

$$\begin{array}{ll} & m \cdot s_{(v,u')} \\ \leqslant & \qquad \{\!\{ \text{ Hypothesis: } [_u\, m\,]^v \in \mathcal{B}^1 \text{ \& Hypothesis (3.7) \& Monotonicity of } \cdot \ \}\!\} \\ & s_{(u,v)} \cdot s_{(v,u')} \\ \leqslant & \qquad \{\!\{ \text{ Hypothesis (3.9) } \}\!\} \\ & s_{(u,u')} \end{array}$$

Inequation (3.13) is proved similarly. For any binary block $[_u\, c\!\downarrow_z \uparrow^w r\,]^v \in \mathcal{B}^2$, we prove that

$$c \cdot s_{(z,w)} \cdot r \cdot s_{(v,u')} \leqslant s_{(u,u')} \ .$$

We are able to prove it.

$$\begin{array}{ll} & c \cdot s_{(z,w)} \cdot r \cdot s_{(v,u')} \\ \leqslant & \qquad \{\!\{ \text{ Hypothesis: } [_u\, c\!\downarrow_z \uparrow^w r\,]^v \in \mathcal{B}^2 \text{ \& Hypothesis (3.8) \& Monotonicity of } \cdot \ \}\!\} \\ & s_{(u,v)} \cdot s_{(v,u')} \\ \leqslant & \qquad \{\!\{ \text{ Hypothesis (3.9) } \}\!\} \\ & s_{(u,u')} \end{array}$$

So, law (3.6) may sometimes require the verification of more hypotheses, but each of them should be simpler and, if not, it is possible to use (3.5). I do not currently know if it is possible to derive (3.6) from (3.5). I do not think it is, but I did not find a counter-example. More on this will be discussed in Section 7.2.

It is interesting to note that (3.6) is committed to a given "direction" (the unfolding always goes from the left to the right). In this particular case, if a specific component solution $s_{(x,y)}$ is considered for an expression $(\!|_x\, \mathcal{B}\, |\!)^y$, it is not always mandatory to have the complete solution vector $\overrightarrow{s}$ for the grammar associated to $(\!|_x\, \mathcal{B}\, |\!)^y$. For example, to prove that

$$(\!|\, \underset{x\,x}{[\,a\,]^{x}}, \underset{y}{[\,b\,]^{y\,x}} \,|\!) \leqslant aa^*$$

for $a, b \in \Sigma_{\mathbf{i}}$, it is not necessary to have the component solutions $s_{(x,y)}$, $s_{(y,x)}$ and $s_{(y,y)}$. Only the component solution $s_{(x,x)} := aa^*$ is mandatory. After all, when unfolding this grammar from the left to the right, only the rewrite rule $[_x\, a\, ]^x$ can be used. This is different from WMVPGs where there could be derivations like

$$P_{(x,x)} \Rightarrow P_{(x,y)}\, P_{(y,x)} \Rightarrow P_{(x,y)}\, P_{(y,y)}\, P_{(y,x)} \Rightarrow P_{(x,y)}\, b\, P_{(y,x)} \ .$$

Of course, this derivation cannot generate a word, but the rewrite rule $[_y\, b\, ]^y$ can be used somewhere. Thus, if a law such as (3.6) is committed to a "direction", there could be fewer cases to check, i.e., some hypotheses could be dropped. It is possible to use this intuition to define a more interesting law than (3.6). The idea is to restrict the inequational system to solve. An inequation can be dropped if it is not used to calculate the actual result desired for a set of rewrite rules $\mathcal{B}$ (committed to a derivation from left-to-right). This idea is represented by a function $\mathsf{F}^*_{\mathcal{B}}$ (called the "forward strategy") that is used in the law. This function approximates the needed component solution. It is the least fixed point of the monotone function $\mathsf{F}^1_{\mathcal{B}} : 2^{V\times V} \to 2^{V\times V}$ defined for all $T \subseteq V \times V$ by:

$$\begin{aligned}\mathsf{F}^1_{\mathcal{B}}(T) := T &\cup \{(y,y') \mid (\exists\ z, m \mid (z,y') \in T : [_z\, m\, ]^y \in \mathcal{B}^1)\} \\ &\cup \{(y,y'),(w,w') \mid (\exists\ z, c, r \mid (z,y') \in T : [_z\, c{\downarrow_w}{\uparrow^{w'}}\, r\, ]^y \in \mathcal{B}^2)\} \ .\end{aligned}$$

Using this function, a more interesting law than (3.6) can be defined. It is taken as the fourth axiom of VPKA. Let $(\!|_x\, \mathcal{B}\, |\!)^y$ be an expression on a finite set of symbols $V$ representing an operator of $\mathcal{G}$ along with its operands. Let $s_{(u,u')}$ be a VPRE for all $(u,u') \in \mathsf{F}^*_{\mathcal{B}}(\{(x,y)\})$. Each $s_{(u,u')}$ acts as a solution for a component (or nonterminal) $P_{(u,u')}$ of the grammar. Then

$$\begin{aligned}&\Big( \wedge\, u, u' \mid (u,u') \in \mathsf{F}^*_{\mathcal{B}}(\{(x,y)\}) : \\ &\qquad (\wedge\, m \mid [\underset{u}{\,m\,}^{u'} \in \mathcal{B}^1 : m \leqslant s_{(u,u')}) \\ &\quad \wedge\ (\wedge\, m, v \mid [\underset{u}{\,m\,}^{v} \in \mathcal{B}^1 : m \cdot s_{(v,u')} \leqslant s_{(u,u')}) \\ &\quad \wedge\ (\wedge\, c, z, r, w \mid [\underset{u}{\,c}{\downarrow_{\underset{z}{}}}{\uparrow^{w}}\, r\, ]^{u'} \in \mathcal{B}^2 : c \cdot s_{(z,w)} \cdot r \leqslant s_{(u,u')}) \\ &\quad \wedge\ (\wedge\, c, z, r, w, v \mid [\underset{u}{\,c}{\downarrow_{\underset{z}{}}}{\uparrow^{w}}\, r\, ]^{v} \in \mathcal{B}^2 : c \cdot s_{(z,w)} \cdot r \cdot s_{(v,u')} \leqslant s_{(u,u')})\Big) \\ &\to\ (\!|\underset{x}{\,\mathcal{B}\,}|\!)^{y} \leqslant s_{(x,y)} \ .\end{aligned} \tag{3.14}$$

Axiom (3.14) easily allows one to derive (3.6). So, law (3.5) can be derived by taking (3.14) as an axiom. Recall that (3.14) is committed to a derivation from *left-to-right* of the grammar. It is also interesting to have an axiom like (3.14), but such that it

is committed to a derivation from *right-to-left* of the grammar. To do this, one can use grammars restricted to well-matched VPLs using a backward strategy from Section 3.1.1 and recall that WMVPGs are equivalent to them. So, these grammars can be taken and an inequational system can be extracted from it (as it was done with WMVPGs). Once again, the inequational system to solve can be restricted. An inequation can be dropped if it is not used to calculate the actual result desired for a set of rewrite rules $\mathcal{B}$ (committed to a derivation from right-to-left). This idea is represented by a function $\mathsf{B}^*_{\mathcal{B}}$ (called the "backward strategy") that is used in the next axiom. This function approximates the needed component solution. It is the least fixed point of the monotone function $\mathsf{B}^1_{\mathcal{B}} : 2^{V\times V} \to 2^{V\times V}$ defined for any $T \subseteq V \times V$ by:

$$\begin{aligned}\mathsf{B}^1_{\mathcal{B}}(T) := T &\cup \{(y,y') \mid (\exists\, z, m \mid (y,z) \in T : [_{y'}\, m\, ]^z \in \mathcal{B}^1)\} \\ &\cup \{(y,y'),(w,w') \mid (\exists\, z,c,r \mid (y,z) \in T : [_{y'}\, c{\downarrow_w}{\uparrow^{w'}}\, r\, ]^z \in \mathcal{B}^2)\}\ .\end{aligned}$$

Using this function, a new axiom can be defined. It is taken as the fifth axiom of VPKA. Let $(\!|_x\, \mathcal{B}\, |\!)^y$ be an expression on a finite set of symbols $V$ representing an operator of $\mathcal{G}$ along with its operands. Let $s_{(u,u')}$ be a VPRE for all $(u,u') \in \mathsf{B}^*_{\mathcal{B}}(\{(x,y)\})$. Each $s_{(u,u')}$ acts as a solution for a component (or nonterminal) $P_{(u,u')}$ of the grammar. Then

$$\begin{aligned}&\Big(\wedge\, u,u' \mid (u,u') \in \mathsf{B}^*_{\mathcal{B}}(\{(x,y)\}) : \\ &\qquad (\wedge\, m \mid [\underset{u}{\overset{u'}{m}}\,] \in \mathcal{B}^1 : m \leqslant s_{(u,u')}) \\ &\quad \wedge\ (\wedge\, m,v \mid [\underset{v}{\overset{u'}{m}}\,] \in \mathcal{B}^1 : s_{(u,v)} \cdot m \leqslant s_{(u,u')}) \\ &\quad \wedge\ (\wedge\, c,z,r,w \mid [\underset{u}{c}\underset{z}{\downarrow}\overset{w}{\uparrow}\overset{u'}{r}\,] \in \mathcal{B}^2 : c \cdot s_{(z,w)} \cdot r \leqslant s_{(u,u')}) \\ &\quad \wedge\ (\wedge\, c,z,r,w,v \mid [\underset{v}{c}\underset{z}{\downarrow}\overset{w}{\uparrow}\overset{u'}{r}\,] \in \mathcal{B}^2 : s_{(u,v)} \cdot c \cdot s_{(z,w)} \cdot r \leqslant s_{(u,u')})\Big) \\ \to\ &(\!|_x\, \mathcal{B}\, |\!)^y \leqslant s_{(x,y)}\ .\end{aligned} \tag{3.15}$$

Note that the first five axioms of VPKA state that the language represented by a WMVPG is exactly the least solution of the inequational system described by the grammar for the component of its starting nonterminal. Note also that the first five axioms of VPKA are inspired by the axioms of the $^*$ operator of Kleene algebra. There are unfolding axioms for the fixed point and induction axioms to represent the operator as the least solution of an inequation. In the case of the $^*$ operator, there is only one inequation to solve, but, for the operators of $\mathcal{G}$, there are multiple inequations (an inequational system). Since we must use the inequational system as a whole, axioms (3.14) and (3.15) are more complex than the induction axioms of the $^*$ operator.

For the moment, five new axioms have been defined for VPKA. One may wonder whether these five axioms suffice to handle any equality of visibly pushdown regular expressions. In our experiments, axioms (3.2), (3.3), (3.4), (3.14) and (3.15) seemed sufficient to handle lots of equalities of visibly pushdown regular expressions if not all of them. However, we saw some limitations when attempting to show that the axiomatic system of VPKA is complete for valid equations between languages denoted by visibly pushdown regular expressions. To realize this proof (presented in Section 4.2), we will encode visibly pushdown automata by expressions of VPKA. So, we will recreate algebraically some results of visibly pushdown automata theory. The problem occurs when trying to prove that two visibly pushdown automata are bisimilar. In this particular case, it is really useful to have a law such as the bisimulation rule of Kleene algebra: for any expressions $p$, $q$ and $r$,

$$rp = qr \to rp^* = q^*r \ .$$

However, it seems impossible to prove a similar law for the operators of $\mathcal{G}$ using only axioms (3.2), (3.3), (3.4), (3.14) and (3.15). So, we now introduce two more axioms that will allow us to resolve this problem. In this dissertation, they will be used *only* in the completeness proof discussed earlier. No other proof of this dissertation needs them. First note that the bisimulation rule of Kleene algebra could have been split into two *simulation rules*: for any expressions $p$, $q$ and $r$,

$$\begin{aligned} rp \leqslant qr \to rp^* \leqslant q^*r \ , \\ rp \geqslant qr \to rp^* \geqslant q^*r \ . \end{aligned}$$

The next two axioms of VPKA are equational implications called $(\!|\,|\!)$*-simulation axioms* that are inspired by the two previous simulation rules. The goal of these axioms is to verify whether a finite list $\mathcal{B}$ of unary and binary blocks on labels from a finite set $V$ can be simulated by another finite list $\mathcal{C}$ of unary and binary blocks on labels from a finite set $V'$. Like for axioms (3.14) and (3.15), the next two axioms are committed to a given "direction":

- The first of these axiom always unfolds from the left to the right (we call it, "travelling forward") to verify that a given relation is a simulation relation;
- The second of these axiom always unfolds from the right to the left (we call it, "travelling backward") to verify that a given relation is a simulation relation.

To simplify the discussion, consider that a label of a block of $\mathcal{B}$ or $\mathcal{C}$ is a state of a visibly pushdown automaton (this is not exactly true, but it will help the reader to understand

the point). The simulation relation is encoded by a set of expressions like $b_{u_2}^{(u',u)}$, where $u, u' \in V$ and $u_2 \in V'$. An expression $b_{u_2}^{(u',u)}$ will be substituted by the constant 0 or 1 in our uses but it can be any expression of the algebra. Intuitively, an expression $b_{u_2}^{(u',u)}$ indicates if the label (state) $u \in V$ is simulated by the label (state) $u_2 \in V'$ when the *first label (state)* processed just after the last unmatched call is $u' \in V$ (such a $u'$ is useful to show some results like the determinization of visibly pushdown automata).

In the first place, let $(\!|_x \mathcal{B} |\!)^y$ be an expression on a finite set of symbols $V$ representing an operator of $\mathcal{G}$ along with its operands. Looking at this expression from the point of view of visibly pushdown automata, we can say that $\mathcal{B}$ represents the transitions of a visibly pushdown automaton by using the following idea[4]:

- a block $[\!|_u m |\!]^{u'} \in \mathcal{B}^1$ represents a transition from a state $u$ to a state $u'$ labelled with the internal action $m$ (we do not consider the cases $m = 1$ and $m = 0$ in this informal presentation);
- a block $[\!|_u c{\downarrow_z}{\uparrow^w} r |\!]^{u'} \in \mathcal{B}^2$ represents a transition from a state $u$ to a state $z$ labelled with the call action $c$ (and pushing a stack symbol $d \in \Gamma$ on the stack) and a transition from a state $w$ to a state $u'$ (when popping the stack symbol $d$ and) labelled with the return action $r$.

Thus, $(\!|_x \mathcal{B} |\!)^y$ should represent the language accepted by the visibly pushdown automaton when starting in the initial state $x$ and ending in the final state $y$ with an empty stack (since the expression only represents well-matched words).[5]

Now, let $\{(\!|_{x_2} \mathcal{C} |\!)^{y_2'}\}_{y_2' \in V'}$ be a set of expressions on a finite set of symbols $V'$ and on a common finite list of unary and binary blocks $\mathcal{C}$, each representing an operator of $\mathcal{G}$ along with its operands. Looking at this set of expressions from the point of view of visibly pushdown automata, we can say that, for any $Y \subseteq V'$, $(\sum y_2' \mid y_2' \in Y : (\!|_{x_2} \mathcal{C} |\!)^{y_2'})$ represents the language accepted by the visibly pushdown automaton described by $\mathcal{C}$ when starting in the state $x_2$ and ending in any of the final states $Y$ with an empty stack.

Next, let $b_{u_2}^{(u',u)}$ be a VPRE for all $u, u' \in V$ and $u_2 \in V'$. Each $b_{u_2}^{(u',u)}$ acts as an indication of whether the label (state) $u \in V$ is simulated by a label (state) $u_2 \in V'$ when the *first label (state)* processed just after the last unmatched call is $u' \in V$. Collectively, the $b_{u_2}^{(u',u)}$ act as a simulation relation. Intuitively, the sixth axiom of

[4] This is not always true. Again, it is just a way to simplify things for the discussion.

[5] Again, this is not always true.

VPKA is an equational implication whose conclusion part states that the automaton described by $(\!|_x \mathcal{B} |\!)^y$ is simulated by an automaton, described by an expression such as $(\sum y_2' \mid y_2' \in Y : (\!|_{x_2} \mathcal{C} |\!)^{y_2'})$, whose initial state (label) $x_2$ simulates the initial state (label) $x$ of the automaton and whose accepting states are exactly the states (labels) that simulate the accepting state (label) $y$. In other words, the conclusion part of the axiom is

$$b_{x_2}^{(y',x)} \cdot \underset{x}{(\!|} \mathcal{B} \overset{y}{|\!)} \leqslant (\sum y_2' \mid y_2' \in V' : \underset{x_2}{(\!|} \mathcal{C} \overset{y_2'}{|\!)} \cdot b_{y_2'}^{(y',y)}) \ .$$

The hypothesis part of the sixth axiom just ensures that the $b_{u_2}^{(u',u)}$ act collectively as a simulation relation. A simulation relation is correct if it preserves inequalities with internal actions, calls and returns for all blocks of the list. Thus, the axiom first ensures that for all $u, u', u'' \in V$ and all $x_2' \in V'$ such that the state (label) $u$ is simulated by the state (label) $x_2'$ when the first state (label) processed just after the last unmatched call is $u''$, and for all transitions $(u, m, \lambda; u', \lambda)$ of an internal action $m$, there exists at least one state (label) $y_2'' \in V'$ such that $(x_2', m, \lambda; y_2'', \lambda)$ exists and the state (label) $u'$ is simulated by the state (label) $y_2''$ when the first state (label) processed just after the last unmatched call is $u''$. In other words, it verifies whether the following formula is valid:

$$\Big( \wedge \ u, u', u'', x_2', m \mid u'' \in V \wedge x_2' \in V' \wedge \underset{u}{[} m \overset{u'}{]} \in \mathcal{B}^1 : \\ b_{x_2'}^{(u'',u)} \cdot m \leqslant (\sum y_2'' \mid \underset{x_2'}{[} m \overset{y_2''}{]} \in \mathcal{C}^1 : m \cdot b_{y_2''}^{(u'',u')}) \Big)$$

Second, it ensures that for all $u, u', u'', z, w \in V$ and all $x_2' \in V'$ such that the state (label) $u$ is simulated by the state (label) $x_2'$ when the first state (label) processed just after the last unmatched call is $u''$, and for all pairs of transitions $(u, c, \lambda; z, d)$ and $(w, r, d; u', \lambda)$ of a call action $c$ and a return action $r$ by using a common stack element $d$, there exists at least states (labels) $z', w', y_2'' \in V'$ such that $(x_2', c, \lambda; z', d')$ and $(w', r, d'; y_2'', \lambda)$ exist for a stack element $d'$, and the state (label) $z$ is simulated by the state (label) $z'$ when the first state (label) processed just after the last unmatched call is $z$ ($z$ is used here because the last unmatched call is the $c$ that was just handled). In other words, it verifies whether the following formula is valid:

$$\Big( \wedge \ u, u', u'', x_2', c, z, r, w \mid u'' \in V \wedge x_2' \in V' \wedge \underset{u}{[} c \underset{z}{\downarrow} \overset{w}{\uparrow} r \overset{u'}{]} \in \mathcal{B}^2 : \\ b_{x_2'}^{(u'',u)} \cdot c \leqslant (\sum z', w', y_2'' \mid \underset{x_2'}{[} c \underset{z'}{\downarrow} \overset{w'}{\uparrow} r \overset{y_2''}{]} \in \mathcal{C}^2 : b_{x_2'}^{(u'',u)} \cdot c \cdot b_{z'}^{(z,z)} \cdot b_{z'}^{(z,z)}) \Big) \ .$$

Lastly, it ensures that for all $u, u', u'', z, w \in V$ and all $x_2', z', w', w'', y_2'' \in V'$ such that

- the state (label) $u$ is simulated by the state (label) $x_2'$ when the first state (label) processed just after the last unmatched call is $u''$,
- the state (label) $z$ is simulated by the state (label) $z'$ when the first state (label) processed just after the last unmatched call is $z$,
- the state (label) $w$ is simulated by the state (label) $w''$ when the first state (label) processed just after the last unmatched call is $z$,

and

- for all pairs of transitions $(u, c, \lambda; z, d)$ and $(w, r, d; u', \lambda)$ of a call action $c$ and a return action $r$ by using a common stack element $d$,
- for all pairs of transitions $(x_2', c, \lambda; z', d')$ and $(w', r, d'; y_2'', \lambda)$ of a call action $c$ and a return action $r$ by using a common stack element $d'$,

there exists at least a state (label) $y_2''' \in V'$ such that $(x_2', c, \lambda; z', d'')$ and $(w'', r, d''; y_2''', \lambda)$ exist for a stack element $d''$, and the state (label) $u'$ is simulated by the state (label) $y_2'''$ when the first state (label) processed just after the last unmatched call is $u''$. In other words, it verifies whether the following formula is valid:

$$\Big( \wedge\ u, u', u'', x_2', c, z, r, w, z', w'' \mid u'' \in V \wedge x_2', z', w'' \in V' \wedge [\underset{u}{c}\downarrow\overset{w}{\underset{z}{\uparrow}} r\overset{u'}{]} \in \mathcal{B}^2 : \\ (\sum w', y_2'' \mid [\underset{x_2'}{c}\downarrow\overset{w'}{\underset{z'}{\uparrow}} r\overset{y_2''}{]} \in \mathcal{C}^2 : b_{x_2'}^{(u'',u)} \cdot c \cdot b_{z'}^{(z,z)} \cdot \underset{z'}{(\!|\,\mathcal{C}\,|\!)}^{w''} \cdot b_{w''}^{(z,w)} \cdot r) \\ \leqslant\ (\sum y_2''' \mid [\underset{x_2'}{c}\downarrow\overset{w''}{\underset{z'}{\uparrow}} r\overset{y_2'''}{]} \in \mathcal{C}^2 : c \cdot \underset{z'}{(\!|\,\mathcal{C}\,|\!)}^{w''} \cdot r \cdot b_{y_2'''}^{(u'',u')})\Big)\ .$$

Note that this set of hypotheses is particular: any label simulated just before and after a call must be remembered until its matching return is reached. This is to remember sufficient information when it comes to simulating the return action.

In essence, if all the hypotheses are valid, then it can be said that the $b_{u_2}^{(u',u)}$ act collectively as a simulation relation for visibly pushdown automata. Thus, it can be concluded that (this is the conclusion part of the sixth axiom) the automaton described by $(\!|_x\, \mathcal{B}\,|\!)^y$ is simulated by an automaton, described by an expression such as $(\sum y_2' \mid y_2' \in Y : (\!|_{x_2}\, \mathcal{C}\,|\!)^{y_2'})$, where its initial state (label) $x_2$ simulates the initial state (label) $x$ of the automaton and its accepting states are exactly the states (labels) that simulate

the accepting state (label) $y$. As a result, the sixth axiom of VPKA is

$$
\begin{aligned}
&\Big( \wedge\ u, u', u'', x_2', m \mid u'' \in V \wedge x_2' \in V' \wedge [\!\!\underset{u}{\overset{u'}{\ }}\!\!] m \in \mathcal{B}^1 : \\
&\quad b_{x_2'}^{(u'',u)} \cdot m \leqslant (\sum\ y_2'' \mid [\!\!\underset{x_2'}{\overset{y_2''}{\ }}\!\!] m \in \mathcal{C}^1 : m \cdot b_{y_2''}^{(u'',u')})\Big) \\
\wedge &\Big( \wedge\ u, u', u'', x_2', c, z, r, w \mid u'' \in V \wedge x_2' \in V' \wedge [\!\!\underset{u}{\overset{u'}{\ }}\!\!] c \underset{z}{\downarrow} \overset{w}{\uparrow} r \in \mathcal{B}^2 : \\
&\quad b_{x_2'}^{(u'',u)} \cdot c \leqslant (\sum\ z', w', y_2'' \mid [\!\!\underset{x_2'}{\overset{y_2''}{\ }}\!\!] c \underset{z'}{\downarrow} \overset{w'}{\uparrow} r \in \mathcal{C}^2 : b_{x_2'}^{(u'',u)} \cdot c \cdot b_{z'}^{(z,z)} \cdot b_{z'}^{(z,z)})\Big) \\
\wedge &\Big( \wedge\ u, u', u'', x_2', c, z, r, w, z', w'' \mid u'' \in V \wedge x_2', z', w'' \in V' \wedge [\!\!\underset{u}{\overset{u'}{\ }}\!\!] c \underset{z}{\downarrow} \overset{w}{\uparrow} r \in \mathcal{B}^2 : \\
&\quad (\sum\ w', y_2'' \mid [\!\!\underset{x_2'}{\overset{y_2''}{\ }}\!\!] c \underset{z'}{\downarrow} \overset{w'}{\uparrow} r \in \mathcal{C}^2 : b_{x_2'}^{(u'',u)} \cdot c \cdot b_{z'}^{(z,z)} \cdot (\!|\underset{z'}{\overset{w''}{\ }}\!|\, \mathcal{C}) \cdot b_{w''}^{(z,w)} \cdot r) \\
&\quad \leqslant\ (\sum\ y_2'' \mid [\!\!\underset{x_2'}{\overset{y_2''}{\ }}\!\!] c \underset{z'}{\downarrow} \overset{w''}{\uparrow} r \in \mathcal{C}^2 : c \cdot (\!|\underset{z'}{\overset{w''}{\ }}\!|\, \mathcal{C}) \cdot r \cdot b_{y_2''}^{(u'',u')})\Big) \\
\to\ & b_{x_2}^{(y',x)} \cdot (\!|\underset{x}{\overset{y}{\ }}\!|\, \mathcal{B}) \leqslant (\sum\ y_2' \mid y_2' \in V' : (\!|\underset{x_2}{\overset{y_2'}{\ }}\!|\, \mathcal{C}) \cdot b_{y_2'}^{(y',y)})\ .
\end{aligned}
\tag{3.16}
$$

The seventh, and last, axiom of VPKA is similar to (3.16), but the simulation is done by travelling "backward". Thus, this axiom is an equational implication whose conclusion part states that the automaton described by an expression $(\!|_{x_2}\, \mathcal{C}\, |\!)^{y_2}$ is simulated by an automaton, described by an expression such as $(\sum\ x' \mid x' \in X : (\!|_{x'}\, \mathcal{B}\, |\!)^{y})$, whose accepting state (label) $y$ simulates the accepting state (label) $y_2$ of the automaton and whose initial states are exactly the states (labels) that simulate the initial state (label) $x_2$. In other words, the conclusion part of the axiom is

$$
(\!|\underset{x_2}{\overset{y_2}{\ }}\!|\, \mathcal{C}) \cdot b_{y_2}^{(y',y)} \leqslant (\sum\ x' \mid x' \in V : b_{x_2}^{(y',x')} \cdot (\!|\underset{x'}{\overset{y}{\ }}\!|\, \mathcal{B}))\ .
$$

Accordingly, the hypotheses of the seventh axiom ensure that the $b_{u_2}^{(u',u)}$ act collectively as a simulation relation for visibly pushdown automata. As for the conclusion part, the definition of a simulation relation is also mirrored in the hypotheses (modulo some

small modifications). Thus, the seventh axiom of VPKA is

$$
\begin{aligned}
&\Big( \wedge\ u,u',x_2',y_2',m \mid u,u' \in V \wedge \underset{x_2'}{[}\ m\ \overset{y_2'}{]} \in \mathcal{C}^1 : \\
&\qquad m \cdot b_{y_2'}^{(u,u')} \leqslant (\textstyle\sum\ u'' \mid \underset{u''}{[}\ m\ \overset{u'}{]} \in \mathcal{B}^1 : b_{x_2'}^{(u,u'')} \cdot m)\Big) \\
\wedge\ &\Big( \wedge\ u,u',x_2',y_2',c,z,r,w \mid u,u' \in V \wedge \underset{x_2'}{[}\ c \underset{z}{\downarrow} \overset{w}{\uparrow} r\ \overset{y_2'}{]} \in \mathcal{C}^2 : \\
&\qquad r \cdot b_{y_2'}^{(u,u')} \leqslant (\textstyle\sum\ u'',z',w' \mid \underset{u''}{[}\ c \underset{z'}{\downarrow} \overset{w'}{\uparrow} r\ \overset{u'}{]} \in \mathcal{B}^2 : b_w^{(z',w')} \cdot b_{x_2'}^{(u,u'')} \cdot r)\Big) \\
\wedge\ &\Big( \wedge\ u,u',x_2',y_2',c,z,r,w,u'',z',w' \mid u \in V \wedge \underset{x_2'}{[}\ c \underset{z}{\downarrow} \overset{w}{\uparrow} r\ \overset{y_2'}{]} \in \mathcal{C}^2 \\
&\qquad \wedge \underset{u''}{[}\ c \underset{z'}{\downarrow} \overset{w'}{\uparrow} r\ \overset{u'}{]} \in \mathcal{B}^2 : (\textstyle\sum\ z'' \mid z'' \in V : c \cdot b_z^{(z',z'')} \cdot \underset{z''}{(\!|}\ \mathcal{B}\ \overset{w'}{|\!)} \cdot b_{x_2'}^{(u,u'')} \cdot r) \\
&\qquad\qquad \leqslant\ b_{x_2'}^{(u,u'')} \cdot c \cdot \underset{z'}{(\!|}\ \mathcal{B}\ \overset{w'}{|\!)} \cdot r\Big) \\
\rightarrow\ &\underset{x_2}{(\!|}\ \mathcal{C}\ \overset{y_2}{|\!)} \cdot b_{y_2}^{(y',y)} \leqslant (\textstyle\sum\ x' \mid x' \in V : b_{x_2}^{(y',x')} \cdot \underset{x'}{(\!|}\ \mathcal{B}\ \overset{y}{|\!)})\ .
\end{aligned}
\tag{3.17}
$$

We now have all the axioms needed to define the algebraic system of visibly pushdown Kleene algebra. Of course, all these axioms are sound for the language model under interpretation $\mathcal{L}$, as shown in Section 4.1.

**Definition 3.12** (Visibly pushdown Kleene algebra (VPKA))**.** Let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be disjoint finite sets of atomic elements such that at least one of the sets is nonempty. A visibly pushdown Kleene algebra is a structure $(K,+,\cdot,^*,0,1,\mathcal{G})$ generated by $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ under the axioms of Kleene algebra (in other words, the structure $(K,+,\cdot,^*,0,1)$ is a Kleene algebra) and such that the laws (3.2), (3.3), (3.4), (3.14), (3.15), (3.16) and (3.17) hold for the structure $(K,+,\cdot,^*,0,1,\mathcal{G})$. ♦

To get a better grip on the axioms, we prove a simple theorem

$$
\underset{x\,x}{(\!|}\ [\,\mathsf{c}\underset{x}{\downarrow}\overset{y}{\uparrow}\mathsf{r}\,]\overset{y}{},\ \underset{x}{[}\,\mathsf{a}\,]\overset{y\,y}{|\!)} \leqslant \underset{x\,x}{(\!|}\ [\,\mathsf{c}\underset{x}{\downarrow}\overset{y}{\uparrow}\mathsf{r}\,]\overset{y}{},\ \underset{x}{[}\,\mathsf{a}\,]\overset{x}{},\ \underset{x}{[}\,\mathsf{a}\,]\overset{y\,y}{|\!)}
$$

for $\Sigma_{\mathbf{i}} := \{\mathsf{a}\}$, $\Sigma_{\mathbf{c}} := \{\mathsf{c}\}$, $\Sigma_{\mathbf{r}} := \{\mathsf{r}\}$ and $V := \{\mathsf{x},\mathsf{y}\}$. Let

$$
\mathcal{C}_{\mathsf{ex}} := \underset{x\ x}{[}\,\mathsf{c}\underset{x}{\downarrow}\overset{y}{\uparrow}\mathsf{r}\,]\overset{y}{},\ \underset{x}{[}\,\mathsf{a}\,]\overset{x}{},\ \underset{x}{[}\,\mathsf{a}\,]\overset{y}{}\ .
$$

By

$$
\mathsf{F}^*_{[_{\mathsf{x}}\,\mathsf{c}\downarrow_{\mathsf{x}}\uparrow^{\mathsf{y}}\,\mathsf{r}\,]^{\mathsf{y}},\,[_{\mathsf{x}}\,\mathsf{a}\,]^{\mathsf{y}}}(\{(\mathsf{x},\mathsf{y})\}) = \{(\mathsf{x},\mathsf{y}),(\mathsf{y},\mathsf{y})\}
$$

and axiom (3.14), it suffices to prove, for $s_{(\mathsf{x},\mathsf{y})} := (\!|_{\mathsf{x}}\, \mathcal{C}_{\mathsf{ex}}\, |\!)^{\mathsf{y}}$ and $s_{(\mathsf{y},\mathsf{y})} := 0$, that

$$\mathsf{a} \leqslant \underset{\mathsf{x}}{(\!|} \mathcal{C}_{\mathsf{ex}} \overset{\mathsf{y}}{|\!)} \quad , \tag{3.18}$$

$$\mathsf{a} \cdot 0 \leqslant \underset{\mathsf{x}}{(\!|} \mathcal{C}_{\mathsf{ex}} \overset{\mathsf{y}}{|\!)} \quad , \tag{3.19}$$

$$\mathsf{c} \cdot \underset{\mathsf{x}}{(\!|} \mathcal{C}_{\mathsf{ex}} \overset{\mathsf{y}}{|\!)} \cdot \mathsf{r} \leqslant \underset{\mathsf{x}}{(\!|} \mathcal{C}_{\mathsf{ex}} \overset{\mathsf{y}}{|\!)} \quad , \tag{3.20}$$

$$\mathsf{c} \cdot \underset{\mathsf{x}}{(\!|} \mathcal{C}_{\mathsf{ex}} \overset{\mathsf{y}}{|\!)} \cdot \mathsf{r} \cdot 0 \leqslant \underset{\mathsf{x}}{(\!|} \mathcal{C}_{\mathsf{ex}} \overset{\mathsf{y}}{|\!)} \quad . \tag{3.21}$$

Inequations (3.19) and (3.21) are trivial by Kleene algebra. Inequations (3.18) and (3.20) are solved respectively by using axioms (3.2) and (3.3).

*Remark* 3.13 (Axiomatic system of visibly pushdown Kleene algebra). Any visibly pushdown Kleene algebra is a partial algebra because each operator of $\mathcal{G}$ is defined only on atomic elements. However, a deductive system for equational logic [47] can be used for visibly pushdown Kleene algebra modulo a slight modification: any substitution of elements of $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ must follow an operator's "typing". This deductive system along with the axioms of visibly pushdown Kleene algebra form the axiomatic system of visibly pushdown Kleene algebra. ⧫

In the sequel, we freely use the classical laws of Kleene algebra (see Section 2.1 or, for example [25]) and the following laws of visibly pushdown Kleene algebra.

### 3.2.2 Some Useful Laws of VPKA

**Some equalities of $(\!|\,|\!)$-expressions:** Let $(\!|_x\, \mathcal{B}\, |\!)^y$ be an expression on a finite set of symbols $V$ representing an operator of $\mathcal{G}$ along with its operands. Here is a theorem of VPKA inspired by the definition of $(\!|_x\, \mathcal{B}\, |\!)^y$ for the block-based definition of visibly pushdown regular expressions:

$$\begin{aligned}
\underset{x}{(\!|}\, \mathcal{B}\, \overset{y}{|\!)} = {} & (\sum m \mid \underset{x}{[}\, m \overset{y}{]} \in \mathcal{B}^1 : m) \\
& + \ (\sum m, v \mid \underset{x}{[}\, m \overset{v}{]} \in \mathcal{B}^1 : m \cdot \underset{v}{(\!|}\, \mathcal{B}\, \overset{y}{|\!)}) \\
& + \ (\sum c, z, r, w \mid \underset{x}{[}\, c \underset{z}{\downarrow} \overset{w}{\uparrow} r \overset{y}{]} \in \mathcal{B}^2 : c \cdot \underset{z}{(\!|}\, \mathcal{B}\, \overset{w}{|\!)} \cdot r) \\
& + \ (\sum c, z, r, w, v \mid \underset{x}{[}\, c \underset{z}{\downarrow} \overset{w}{\uparrow} r \overset{v}{]} \in \mathcal{B}^2 : c \cdot \underset{z}{(\!|}\, \mathcal{B}\, \overset{w}{|\!)} \cdot r \cdot \underset{v}{(\!|}\, \mathcal{B}\, \overset{y}{|\!)}) \ .
\end{aligned} \tag{3.22}$$

The $\geqslant$ part is proved using the KA law

$$p_1 + p_2 \leqslant p_3 \leftrightarrow p_1 \leqslant p_3 \wedge p_2 \leqslant p_3$$

and repeated use of axioms (3.2), (3.3) and (3.4). For the $\leqslant$ part, axiom (3.14) can be used with solutions

$$\begin{aligned} s_{(u,u')} := {} & (\sum m \mid \underset{u}{[}\, m \,\overset{u'}{]} \in \mathcal{B}^1 : m) \\ & + (\sum m, v \mid \underset{u}{[}\, m \,\overset{v}{]} \in \mathcal{B}^1 : m \cdot \underset{v}{(\!|}\, \mathcal{B} \,\overset{u'}{|\!)}) \\ & + (\sum c, z, r, w \mid \underset{u}{[}\, c \underset{z}{\downarrow} \overset{w}{\uparrow} r \,\overset{u'}{]} \in \mathcal{B}^2 : c \cdot \underset{z}{(\!|}\, \mathcal{B} \,\overset{w}{|\!)} \cdot r) \\ & + (\sum c, z, r, w, v \mid \underset{u}{[}\, c \underset{z}{\downarrow} \overset{w}{\uparrow} r \,\overset{v}{]} \in \mathcal{B}^2 : c \cdot \underset{z}{(\!|}\, \mathcal{B} \,\overset{w}{|\!)} \cdot r \cdot \underset{v}{(\!|}\, \mathcal{B} \,\overset{u'}{|\!)}) \ , \end{aligned}$$

for all labels $u, u' \in \mathsf{F}^*_{\mathcal{B}}(\{(x,y)\})$. So, to prove

$$\underset{x}{(\!|}\, \mathcal{B} \,\overset{y}{|\!)} \leqslant s_{(x,y)} \ ,$$

it suffices to prove that, for all labels $u, u' \in \mathsf{F}^*_{\mathcal{B}}(\{(x,y)\})$,

$$\begin{aligned} & (\wedge\, m \mid \underset{u}{[}\, m \,\overset{u'}{]} \in \mathcal{B}^1 : m \leqslant s_{(u,u')}) \\ & \wedge\ (\wedge\, m, v \mid \underset{u}{[}\, m \,\overset{v}{]} \in \mathcal{B}^1 : m \cdot s_{(v,u')} \leqslant s_{(u,u')}) \\ & \wedge\ (\wedge\, c, z, r, w \mid \underset{u}{[}\, c \underset{z}{\downarrow} \overset{w}{\uparrow} r \,\overset{u'}{]} \in \mathcal{B}^2 : c \cdot s_{(z,w)} \cdot r \leqslant s_{(u,u')}) \\ & \wedge\ (\wedge\, c, z, r, w, v \mid \underset{u}{[}\, c \underset{z}{\downarrow} \overset{w}{\uparrow} r \,\overset{v}{]} \in \mathcal{B}^2 : c \cdot s_{(z,w)} \cdot r \cdot s_{(v,u')} \leqslant s_{(u,u')}) \ . \end{aligned}$$

By the KA law $p_1 + p_2 \leqslant p_3 \leftrightarrow p_1 \leqslant p_3 \wedge p_2 \leqslant p_3$, it suffices to prove that, for all labels $u, u' \in \mathsf{F}^*_{\mathcal{B}}(\{(x,y)\})$,

$$\begin{aligned} s_{(u,u')} \geqslant {} & (\sum m \mid \underset{u}{[}\, m \,\overset{u'}{]} \in \mathcal{B}^1 : m) \\ & + (\sum m, v \mid \underset{u}{[}\, m \,\overset{v}{]} \in \mathcal{B}^1 : m \cdot s_{(v,u')}) \\ & + (\sum c, z, r, w \mid \underset{u}{[}\, c \underset{z}{\downarrow} \overset{w}{\uparrow} r \,\overset{u'}{]} \in \mathcal{B}^2 : c \cdot s_{(z,w)} \cdot r) \\ & + (\sum c, z, r, w, v \mid \underset{u}{[}\, c \underset{z}{\downarrow} \overset{w}{\uparrow} r \,\overset{v}{]} \in \mathcal{B}^2 : c \cdot s_{(z,w)} \cdot r \cdot s_{(v,u')}) \ . \end{aligned} \tag{3.23}$$

Since the $\geqslant$ part of (3.22) has been proved, then for all labels $u, u' \in \mathsf{F}^*_{\mathcal{B}}(\{(x,y)\})$,

$$s_{(u,u')} \leqslant \underset{u}{(\!|}\, \mathcal{B} \,\overset{u'}{|\!)} \ . \tag{3.24}$$

Using (3.24) in (3.23) along with the transitivity of $\leqslant$ and the monotonicity of $\cdot$ and $+$, it suffices to prove that, for all labels $u, u' \in \mathsf{F}^*_{\mathcal{B}}(\{(x,y)\})$,

$$\begin{aligned} s_{(u,u')} \geqslant\ & (\sum m \mid [\,{\scriptstyle u}\, m \,]^{u'} \in \mathcal{B}^1 : m) \\ & + (\sum m, v \mid [\,{\scriptstyle u}\, m \,]^{v} \in \mathcal{B}^1 : m \cdot (\!|\,{\scriptstyle v}\, \mathcal{B} \,{\scriptstyle u'}\,|\!)) \\ & + (\sum c, z, r, w \mid [\,{\scriptstyle u}\, c \downarrow_{z} \uparrow^{w} r \,]^{u'} \in \mathcal{B}^2 : c \cdot (\!|\,{\scriptstyle z}\, \mathcal{B} \,{\scriptstyle w}\,|\!) \cdot r) \\ & + (\sum c, z, r, w, v \mid [\,{\scriptstyle u}\, c \downarrow_{z} \uparrow^{w} r \,]^{v} \in \mathcal{B}^2 : c \cdot (\!|\,{\scriptstyle z}\, \mathcal{B} \,{\scriptstyle w}\,|\!) \cdot r \cdot (\!|\,{\scriptstyle v}\, \mathcal{B} \,{\scriptstyle u'}\,|\!)) \ . \end{aligned}$$

This is trivial by the definition of $s_{(u,u')}$ (reflexivity of $\leqslant$).

Of course, a similar theorem holds for the "backward travelling" version of (3.22):

$$\begin{aligned} (\!|\,{\scriptstyle x}\, \mathcal{B} \,{\scriptstyle y}\,|\!) = & (\sum m \mid [\,{\scriptstyle x}\, m \,]^{y} \in \mathcal{B}^1 : m) \\ & + (\sum m, v \mid [\,{\scriptstyle v}\, m \,]^{y} \in \mathcal{B}^1 : (\!|\,{\scriptstyle x}\, \mathcal{B} \,{\scriptstyle v}\,|\!) \cdot m) \\ & + (\sum c, z, r, w \mid [\,{\scriptstyle x}\, c \downarrow_{z} \uparrow^{w} r \,]^{y} \in \mathcal{B}^2 : c \cdot (\!|\,{\scriptstyle z}\, \mathcal{B} \,{\scriptstyle w}\,|\!) \cdot r) \\ & + (\sum c, z, r, w, v \mid [\,{\scriptstyle v}\, c \downarrow_{z} \uparrow^{w} r \,]^{y} \in \mathcal{B}^2 : (\!|\,{\scriptstyle x}\, \mathcal{B} \,{\scriptstyle v}\,|\!) \cdot c \cdot (\!|\,{\scriptstyle z}\, \mathcal{B} \,{\scriptstyle w}\,|\!) \cdot r) \ . \end{aligned} \tag{3.25}$$

The proof is similar to (3.22), but uses (3.15) instead of (3.14).

**Add/Remove a unary block:** Let $(\!|_x\, \mathcal{B}, [_y\, a\,]^{y'}, \mathcal{C}\,|\!)^{x'}$ be an expression on a finite set of symbols $V$ representing an operator of $\mathcal{G}$ along with its operands. Then,

$$(\!|\,{\scriptstyle x}\, \mathcal{B}, \mathcal{C} \,{\scriptstyle x'}\,|\!) \leqslant (\!|\,{\scriptstyle x}\, \mathcal{B}, [\,{\scriptstyle y}\, a \,]^{y'}, \mathcal{C} \,{\scriptstyle x'}\,|\!) \ . \tag{3.26}$$

To prove (3.26), we use (3.5) with $s_{(u,u')} := (\!|_u\, \mathcal{B}, [_y\, a\,]^{y'}, \mathcal{C}\,|\!)^{u'}$ for all $u, u' \in V$. So, it suffices to prove that, for all $u, u' \in V$,

$$(\wedge\, m \mid [\,{\scriptstyle u}\, m \,]^{u'} \in (\mathcal{B}, \mathcal{C})^1 : m \leqslant (\!|\,{\scriptstyle u}\, \mathcal{B}, [\,{\scriptstyle y}\, a \,]^{y'}, \mathcal{C} \,{\scriptstyle u'}\,|\!)) \ , \tag{3.27}$$

$$(\wedge\, c, z, r, w \mid [\,{\scriptstyle u}\, c \downarrow_{z} \uparrow^{w} r \,]^{u'} \in (\mathcal{B}, \mathcal{C})^2 : c \cdot (\!|\,{\scriptstyle z}\, \mathcal{B}, [\,{\scriptstyle y}\, a \,]^{y'}, \mathcal{C} \,{\scriptstyle w}\,|\!) \cdot r \leqslant (\!|\,{\scriptstyle u}\, \mathcal{B}, [\,{\scriptstyle y}\, a \,]^{y'}, \mathcal{C} \,{\scriptstyle u'}\,|\!)) \ , \tag{3.28}$$

$$(\wedge\, v \mid v \in V : (\!|\,{\scriptstyle u}\, \mathcal{B}, [\,{\scriptstyle y}\, a \,]^{y'}, \mathcal{C} \,{\scriptstyle v}\,|\!) \cdot (\!|\,{\scriptstyle v}\, \mathcal{B}, [\,{\scriptstyle y}\, a \,]^{y'}, \mathcal{C} \,{\scriptstyle u'}\,|\!) \leqslant (\!|\,{\scriptstyle u}\, \mathcal{B}, [\,{\scriptstyle y}\, a \,]^{y'}, \mathcal{C} \,{\scriptstyle u'}\,|\!)) \ . \tag{3.29}$$

Inequations (3.27), (3.28) and (3.29) are proved by Kleene algebra, set theory and respectively by axioms (3.2), (3.3) and (3.4).

**Add/Remove a binary block:** Let $(\!|_x \mathcal{B}, [_y\, c \downarrow_{y_1} \uparrow^{y_2} r\,]^{y'}, \mathcal{C} |\!)^{x'}$ be an expression on a finite set of symbols $V$ representing an operator of $\mathcal{G}$ along with its operands. Then,

$$(\!|_x \mathcal{B}, \mathcal{C} |\!)^{x'} \leqslant (\!|_x \mathcal{B}, [_y\, c \downarrow_{y_1} \uparrow^{y_2} r\,]^{y'}, \mathcal{C} |\!)^{x'} \quad . \tag{3.30}$$

The proof is similar to (3.26).

**Idempotency of blocks:** Let $(\!|_x \mathcal{B}, [_y\, a\,]^{y'}, \mathcal{C} |\!)^{x'}$ and $(\!|_x \mathcal{B}, [_y\, c \downarrow_z \uparrow^{z'} r\,]^{y'}, \mathcal{C} |\!)^{x'}$ be expressions on a finite set of symbols $V$ representing operators of $\mathcal{G}$ along with their operands. Then,

$$(\!|_x \mathcal{B}, [_y\, a\,]^{y'}, [_y\, a\,]^{y'}, \mathcal{C} |\!)^{x'} = (\!|_x \mathcal{B}, [_y\, a\,]^{y'}, \mathcal{C} |\!)^{x'} \quad , \tag{3.31}$$

$$(\!|_x \mathcal{B}, [_y\, c \downarrow_z \uparrow^{z'} r\,]^{y'}, [_y\, c \downarrow_z \uparrow^{z'} r\,]^{y'}, \mathcal{C} |\!)^{x'} = (\!|_x \mathcal{B}, [_y\, c \downarrow_z \uparrow^{z'} r\,]^{y'}, \mathcal{C} |\!)^{x'} \quad . \tag{3.32}$$

The case $\geqslant$ is immediate from (3.26) and (3.30). The case $\leqslant$ is immediate from (3.14) with

$$s_{(u,u')} := (\!|_u \mathcal{B}, [_y\, a\,]^{y'}, \mathcal{C} |\!)^{u'} \text{ for (3.31)} \qquad \text{and} \qquad s_{(u,u')} := (\!|_u \mathcal{B}, [_y\, c \downarrow_z \uparrow^{z'} r\,]^{y'}, \mathcal{C} |\!)^{u'} \text{ for (3.32)}$$

for all $u, u' \in V$ and noting that, for $i \in \{1, 2\}$,

$$(\mathcal{B}, [_y\, a\,]^{y'}, [_y\, a\,]^{y'}, \mathcal{C})^i = (\mathcal{B}, [_y\, a\,]^{y'}, \mathcal{C})^i$$

and

$$(\mathcal{B}, [_y\, c \downarrow_z \uparrow^{z'} r\,]^{y'}, [_y\, c \downarrow_z \uparrow^{z'} r\,]^{y'}, \mathcal{C})^i = (\mathcal{B}, [_y\, c \downarrow_z \uparrow^{z'} r\,]^{y'}, \mathcal{C})^i \quad .$$

**Swap blocks:** Let

$$(\!|_x \mathcal{B}, [_{y_1}\, a\,]^{y_1'}, [_{y_2}\, b\,]^{y_2'}, \mathcal{C} |\!)^{x'} \quad ,$$

$$(\!|_x \mathcal{B}, [_{y_1}\, a\,]^{y_1'}, [_{y_3}\, c_1 \downarrow_{z_1} \uparrow^{z_1'} r_1\,]^{y_3'}, \mathcal{C} |\!)^{x'} \quad ,$$

$$(\!|_x \mathcal{B}, [_{y_3}\, c_1 \downarrow_{z_1} \uparrow^{z_1'} r_1\,]^{y_3'}, [_{y_4}\, c_2 \downarrow_{z_2} \uparrow^{z_2'} r_2\,]^{y_4'}, \mathcal{C} |\!)^{x'}$$

be expressions on a finite set of symbols $V$ representing operators of $\mathcal{G}$ along with their operands. Then,

$$\mathop{(\!|}_{x} \mathcal{B}, \mathop{[}_{y_1}^{y_1'} a \mathop{]}^{y_1'}, \mathop{[}_{y_2}^{y_2'} b \mathop{]}^{y_2'}, \mathcal{C} \mathop{|\!)}^{x'} = \mathop{(\!|}_{x} \mathcal{B}, \mathop{[}_{y_2}^{y_2'} b \mathop{]}^{y_2'}, \mathop{[}_{y_1}^{y_1'} a \mathop{]}^{y_1'}, \mathcal{C} \mathop{|\!)}^{x'} \ , \tag{3.33}$$

$$\mathop{(\!|}_{x} \mathcal{B}, \mathop{[}_{y_1}^{y_1'} a \mathop{]}^{y_1'}, \mathop{[}_{y_3}^{y_3'} c_1 \mathop{\downarrow}_{z_1}^{z_1'} \mathop{\uparrow} r_1 \mathop{]}^{y_3'}, \mathcal{C} \mathop{|\!)}^{x'} = \mathop{(\!|}_{x} \mathcal{B}, \mathop{[}_{y_3}^{y_3'} c_1 \mathop{\downarrow}_{z_1}^{z_1'} \mathop{\uparrow} r_1 \mathop{]}^{y_3'}, \mathop{[}_{y_1}^{y_1'} a \mathop{]}^{y_1'}, \mathcal{C} \mathop{|\!)}^{x'} \ , \tag{3.34}$$

$$\mathop{(\!|}_{x} \mathcal{B}, \mathop{[}_{y_3}^{y_3'} c_1 \mathop{\downarrow}_{z_1}^{z_1'} \mathop{\uparrow} r_1 \mathop{]}^{y_3'}, \mathop{[}_{y_4}^{y_4'} c_2 \mathop{\downarrow}_{z_2}^{z_2'} \mathop{\uparrow} r_2 \mathop{]}^{y_4'}, \mathcal{C} \mathop{|\!)}^{x'} = \mathop{(\!|}_{x} \mathcal{B}, \mathop{[}_{y_4}^{y_4'} c_2 \mathop{\downarrow}_{z_2}^{z_2'} \mathop{\uparrow} r_2 \mathop{]}^{y_4'}, \mathop{[}_{y_3}^{y_3'} c_1 \mathop{\downarrow}_{z_1}^{z_1'} \mathop{\uparrow} r_1 \mathop{]}^{y_3'}, \mathcal{C} \mathop{|\!)}^{x'} \ . \tag{3.35}$$

The cases $\leqslant$ and $\geqslant$ are direct from (3.14) with appropriate solutions and noting that, for $i \in \{1, 2\}$,

$$\begin{aligned}
&(\mathcal{B}, \mathop{[}_{y_1} a \mathop{]}^{y_1'}, \mathop{[}_{y_2} b \mathop{]}^{y_2'}, \mathcal{C})^i = (\mathcal{B}, \mathop{[}_{y_2} b \mathop{]}^{y_2'}, \mathop{[}_{y_1} a \mathop{]}^{y_1'}, \mathcal{C})^i \ , \\
&(\mathcal{B}, \mathop{[}_{y_1} a \mathop{]}^{y_1'}, \mathop{[}_{y_3} c_1 \mathop{\downarrow}_{z_1} \mathop{\uparrow}^{z_1'} r_1 \mathop{]}^{y_3'}, \mathcal{C})^i = (\mathcal{B}, \mathop{[}_{y_3} c_1 \mathop{\downarrow}_{z_1} \mathop{\uparrow}^{z_1'} r_1 \mathop{]}^{y_3'}, \mathop{[}_{y_1} a \mathop{]}^{y_1'}, \mathcal{C})^i \ , \\
&(\mathcal{B}, \mathop{[}_{y_3} c_1 \mathop{\downarrow}_{z_1} \mathop{\uparrow}^{z_1'} r_1 \mathop{]}^{y_3'}, \mathop{[}_{y_4} c_2 \mathop{\downarrow}_{z_2} \mathop{\uparrow}^{z_2'} r_2 \mathop{]}^{y_4'}, \mathcal{C})^i \\
&\qquad = \ (\mathcal{B}, \mathop{[}_{y_4} c_2 \mathop{\downarrow}_{z_2} \mathop{\uparrow}^{z_2'} r_2 \mathop{]}^{y_4'}, \mathop{[}_{y_3} c_1 \mathop{\downarrow}_{z_1} \mathop{\uparrow}^{z_1'} r_1 \mathop{]}^{y_3'}, \mathcal{C})^i \ .
\end{aligned}$$

Note that the *swap blocks* and the *idempotency of blocks* laws (laws (3.31) to (3.35)) are easy and intuitive laws of VPKA. So, their uses are not mentioned in the remaining proofs.

**Add/Remove a unary block containing** 0**:** Let $\mathop{(\!|}_{x} \mathcal{B}, \mathop{[}_{y} 0 \mathop{]}^{y'} \mathop{|\!)}^{x'}$ be an expression on a finite set of symbols $V$ representing an operator of $\mathcal{G}$ along with its operands. Then,

$$\mathop{(\!|}_{x} \mathcal{B}, \mathop{[}_{y}^{y'} 0 \mathop{]}^{x'} \mathop{|\!)} = \mathop{(\!|}_{x} \mathcal{B} \mathop{|\!)}^{x'} \ . \tag{3.36}$$

The case $\geqslant$ is direct from (3.26). The case $\leqslant$ is direct from (3.14) with $s_{(u,u')} := \mathop{(\!|}_{u} \mathcal{B} \mathop{|\!)}^{u'}$ for all $u, u' \in V$ and using zero of $\cdot$, identity of $+$ and some simple quantification laws.

**Substitution function:** Let $\mathop{(\!|}_{x} \mathcal{B} \mathop{|\!)}^{x'}$ be an expression on a finite set of symbols $V$ representing an operator of $\mathcal{G}$ along with its operands. Let $V'$ be another finite set of symbols. Let $f : V \to V'$ be a total function. Define $\widehat{f}$ to be the trivial extension of the function $f$ to lists of unary and binary blocks (this function only modifies every label

of a block according to $f$). Then,

$$\mathop{(\!|}_{x}\, \mathcal{B}\, \mathop{|\!)}^{x'} \leqslant \mathop{(\!|}_{f(x)}\, \widehat{f}(\mathcal{B})\, \mathop{|\!)}^{f(x')} \quad . \tag{3.37}$$

To prove it, we use (3.5) with $s_{(u,u')} := (\!|_{f(u)}\, \widehat{f}(\mathcal{B})\, |\!)^{f(u')}$ for all $u, u' \in V$. So, it suffices to prove that, for all $u, u' \in V$,

$$(\wedge\, m \mid \mathop{[\, m\, ]}_{u}^{u'} \in \mathcal{B}^1 : m \leqslant \mathop{(\!|}_{f(u)}\, \widehat{f}(\mathcal{B})\, \mathop{|\!)}^{f(u')}) \ , \tag{3.38}$$

$$(\wedge\, c, z, r, w \mid \mathop{[\, c\!\downarrow}_{u}\!\!\mathop{\uparrow\! r\, ]}_{z}^{w\quad u'} \in \mathcal{B}^2 : c \cdot \mathop{(\!|}_{f(z)}\, \widehat{f}(\mathcal{B})\, \mathop{|\!)}^{f(w)} \cdot r \leqslant \mathop{(\!|}_{f(u)}\, \widehat{f}(\mathcal{B})\, \mathop{|\!)}^{f(u')}) \ , \tag{3.39}$$

$$(\wedge\, v \mid v \in V : \mathop{(\!|}_{f(u)}\, \widehat{f}(\mathcal{B})\, \mathop{|\!)}^{f(v)} \cdot \mathop{(\!|}_{f(v)}\, \widehat{f}(\mathcal{B})\, \mathop{|\!)}^{f(u')} \leqslant \mathop{(\!|}_{f(u)}\, \widehat{f}(\mathcal{B})\, \mathop{|\!)}^{f(u')}) \ . \tag{3.40}$$

By definition of $f$ and $\widehat{f}$, $[_u\, m\, ]^{u'} \in \mathcal{B}^1 \Rightarrow [_{f(u)}\, m\, ]^{f(u')} \in (\widehat{f}(\mathcal{B}))^1$ and $[_u\, c\!\downarrow_z\!\uparrow^w r\, ]^{u'} \in \mathcal{B}^2 \Rightarrow [_{f(u)}\, c\!\downarrow_{f(z)}\!\uparrow^{f(w)} r\, ]^{f(u')} \in (\widehat{f}(\mathcal{B}))^2$, and $v \in V \Rightarrow f(v) \in V'$. So, inequations (3.38), (3.39) and (3.40) follow respectively from axioms (3.2), (3.3) and (3.4).

### 3.2.3 Comparison of VPKA with Other Axiomatizations of Subclasses of Context-Free Languages

The axiomatization of subclasses of context-free languages is not new. Leiß proposed Kleene algebra with recursion [36] which is essentially an idempotent semiring with an explicit general least fixed point operator $\mu$. Bloom and Ésik did something similar to Leiß when they defined iteration algebras [5], but they did it in the more general setting of fixed point operators. In contrast, we define a family $\mathcal{G}$ of partial operators (grammar patterns) that are implicit least fixed points and deal only with a restricted set of fixed point formulae.

Note that Leiß also proposed an axiomatization of linear context-free languages through the notion of Kleene modules over a Kleene algebra [37]. However, linear context-free languages and visibly pushdown languages are incomparable.

# Chapter 4

# Soundness, Completeness and Complexity Results for VPKA

This chapter shows that it is possible to automate the process of deciding whether an equation is a theorem of VPKA. This is important since proving equations is at the heart of the process of doing interprocedural program analyses in VPKA.

More explicitly, this chapter shows that the equational theory of visibly pushdown Kleene algebra is EXPTIME-complete. This result is proved by first showing that the axiomatic system of VPKA is sound and complete for valid equations between languages denoted by visibly pushdown regular expressions. In other words, for any visibly pushdown regular expressions $p$ and $q$,

$$\vdash p = q \iff \mathcal{L}(p) = \mathcal{L}(q) \ .$$

## 4.1 Soundness over the Language Model under Interpretation $\mathcal{L}$

We show the following theorem.

**Theorem 4.1.** *The axiomatic system of* VPKA *is sound with respect to the language model under interpretation* $\mathcal{L}$*.*

*Proof.* To show that the axiomatic system is sound, it suffices to verify that all axioms of VPKA are valid in the language model under interpretation $\mathcal{L}$ because it is trivial to

see that each inference rule of the axiomatic system given in Remark 3.13 preserves the validity of formulae. It is known that the algebra $(2^{(\Sigma_{\mathbf{i}} \cup \Sigma_{\mathbf{c}} \cup \Sigma_{\mathbf{r}})^*}, \cup, \bullet, ^*, \emptyset, \{\varepsilon\})$ is a Kleene algebra [25]. So, it suffices to verify that axioms (3.2), (3.3), (3.4), (3.14), (3.15), (3.16) and (3.17) are valid in the language model under interpretation $\mathcal{L}$. We just show that axioms (3.3), (3.4), (3.14) and (3.16) are valid. The other axioms are proved similarly. To ease the proof, we use the block-based definition of VPREs.

**Axiom (3.3) is valid:** We suppose that $[_x\, c \downarrow_z \uparrow^w r\, ]^y \in \mathcal{B}^2$ and we prove that $\{c\} \bullet (\!|_z\, \mathcal{B}\, |\!)^w \bullet \{r\} \subseteq (\!|_x\, \mathcal{B}\, |\!)^y$.

$$
\begin{array}{ll}
 & \{c\} \bullet (\!|_z\, \mathcal{B}\, |\!)^w \bullet \{r\} \\
= & \quad \{\!\{ \text{ Definition of } (\!|_z\, \mathcal{B}\, |\!)^w \text{ for the language model (block-based definition) } \}\!\} \\
 & \{c\} \bullet (\cup\, n \mid n \in \mathbb{N} : (\!|_z\, \mathcal{B}\, |\!)^w_n) \bullet \{r\} \\
= & \quad \{\!\{ \text{ Distributivity of } \bullet \text{ over } \cup \}\!\} \\
 & (\cup\, n \mid n \in \mathbb{N} : \{c\} \bullet (\!|_z\, \mathcal{B}\, |\!)^w_n \bullet \{r\}) \\
\subseteq & \quad \{\!\{ \text{ Hypothesis: } [_x\, c \downarrow_z \uparrow^w r\, ]^y \in \mathcal{B}^2 \text{ \& Set theory \& Definition of } (\!|_x\, \mathcal{B}\, |\!)^y_{n+1} \text{ for} \\
 & \qquad n \in \mathbb{N} \}\!\} \\
 & (\cup\, n \mid n \in \mathbb{N} : (\!|_x\, \mathcal{B}\, |\!)^y_{n+1}) \\
\subseteq & \quad \{\!\{ \text{ Set theory } \}\!\} \\
 & (\cup\, n \mid n \in \mathbb{N} : (\!|_x\, \mathcal{B}\, |\!)^y_{n+1}) \;\cup\; (\!|_x\, \mathcal{B}\, |\!)^y_0 \\
= & \quad \{\!\{ \text{ Arithmetic \& Quantification: Split off term } \}\!\} \\
 & (\cup\, n \mid n \in \mathbb{N} : (\!|_x\, \mathcal{B}\, |\!)^y_n) \\
= & \quad \{\!\{ \text{ Definition of } (\!|_x\, \mathcal{B}\, |\!)^y \text{ for the language model (block-based definition) } \}\!\} \\
 & (\!|_x\, \mathcal{B}\, |\!)^y
\end{array}
$$

**Axiom (3.4) is valid:** We show that, for all words $w$,

$$w \in (\!|\, \underset{x}{\mathcal{B}}\, |\!)^{y'}_{y'} \bullet (\!|\, \underset{y'}{\mathcal{B}}\, |\!)^{y} \Rightarrow w \in (\!|\, \underset{x}{\mathcal{B}}\, |\!)^{y} \quad .$$

We suppose that $w \in (\!|_x\, \mathcal{B}\, |\!)^{y'} \bullet (\!|_{y'}\, \mathcal{B}\, |\!)^y$ and we show that $w \in (\!|_x\, \mathcal{B}\, |\!)^y$. By the hypothesis

$$w \in (\!|\, \underset{x}{\mathcal{B}}\, |\!)^{y'}_{y'} \bullet (\!|\, \underset{y'}{\mathcal{B}}\, |\!)^{y}$$

and the definitions of $(\!|_x\, \mathcal{B}\, |\!)^{y'}$, $(\!|_{y'}\, \mathcal{B}\, |\!)^y$ and $\bullet$ for the language model, there exists at least two words $w_1$ and $w_2$ and numbers $k_1, k_2 \in \mathbb{N}$ such that $w = w_1 w_2$, and $w_1 \in (\!|_x\, \mathcal{B}\, |\!)^{y'}_{k_1}$ and $w_2 \in (\!|_{y'}\, \mathcal{B}\, |\!)^y_{k_2}$. Since $w_1 \in (\!|_x\, \mathcal{B}\, |\!)^{y'}_{k_1}$ (respectively, $w_2 \in (\!|_{y'}\, \mathcal{B}\, |\!)^y_{k_2}$), there exists a

correct travelling $b_1 b_2 \dots b_{k_1+1}$ (respectively, $b'_1 b'_2 \dots b'_{k_2+1}$) of blocks of $\mathcal{B}$ starting with $x$ (respectively, $y'$), ending with $y'$ (respectively, $y$) and producing a set of well-matched words that contains $w_1$ (respectively, $w_2$). Thus, it is possible to concatenate these two correct travellings to generate another correct travelling $b_1 b_2 \dots b_{k_1+1} b'_1 b'_2 \dots b'_{k_2+1}$ of length $k_1 + k_2 + 2$ of blocks of $\mathcal{B}$ starting with $x$, ending with $y$ and producing a set of well-matched words that contains $w_1 w_2$. Using the hypothesis $w = w_1 w_2$ and the definition of $(\!|_x \mathcal{B} |\!)^y_{k_1+k_2+1}$, the previous correct travelling means that $w \in (\!|_x \mathcal{B} |\!)^y_{k_1+k_2+1}$. So, $w \in (\!|_x \mathcal{B} |\!)^y$.

**Axiom** (3.14) **is valid:** We suppose that $x, y \in V$ and that a set of inclusions of the form

$$(\wedge\, m \mid \underset{u}{[}\, m \,\overset{u'}{]} \in \mathcal{B}^1 : \{m\} \subseteq s_{(u,u')}) \ , \tag{4.1}$$

$$(\wedge\, m, v \mid \underset{u}{[}\, m \,\overset{v}{]} \in \mathcal{B}^1 : \{m\} \bullet s_{(v,u')} \subseteq s_{(u,u')}) \ , \tag{4.2}$$

$$(\wedge\, c, z, r, w \mid \underset{u}{[}\, c \underset{z}{\downarrow} \overset{w}{\uparrow} r \,\overset{u'}{]} \in \mathcal{B}^2 : \{c\} \bullet s_{(z,w)} \bullet \{r\} \subseteq s_{(u,u')}) \ , \tag{4.3}$$

$$(\wedge\, c, z, r, w, v \mid \underset{u}{[}\, c \underset{z}{\downarrow} \overset{w}{\uparrow} r \,\overset{v}{]} \in \mathcal{B}^2 : \{c\} \bullet s_{(z,w)} \bullet \{r\} \bullet s_{(v,u')} \subseteq s_{(u,u')}) \ , \tag{4.4}$$

is valid for all $(u, u') \in \mathsf{F}^*_{\mathcal{B}}(\{(x, y)\})$. We show that

$$\underset{x}{(\!|} \mathcal{B} \overset{y}{|\!)} \subseteq s_{(x,y)}$$

is valid. By definition of $(\!|_x \mathcal{B} |\!)^y$ for the language model (block-based definition), it suffices to prove that

$$(\cup\, n \mid n \in \mathbb{N} : \underset{x}{(\!|} \mathcal{B} \overset{y}{\underset{n}{|\!)}}) \subseteq s_{(x,y)} \ .$$

By set theory, it suffices to prove that

$$\underset{x}{(\!|} \mathcal{B} \overset{y}{\underset{n}{|\!)}} \subseteq s_{(x,y)}$$

for all $n \in \mathbb{N}$. A more general result is proved: for all $(u, u') \in \mathsf{F}^*_{\mathcal{B}}(\{(x, y)\})$ and $n \in \mathbb{N}$,

$$\underset{u}{(\!|} \mathcal{B} \overset{u'}{\underset{n}{|\!)}} \subseteq s_{(u,u')} \ .$$

This proof is done by generalized induction over $n$. For the base case ($n = 0$), we show that

$$\underset{u}{(\!|} \mathcal{B} \overset{u'}{\underset{0}{|\!)}} \subseteq s_{(u,u')} \ .$$

By definition of $(\!|_u \mathcal{B} |\!)^{u'}_0$, it suffices to show that

$$(\cup\, m \mid \underset{u}{[\, m \,]}^{u'} \in \mathcal{B}^1 : \{m\}) \subseteq s_{(u,u')} \ .$$

By set theory, it suffices to show that

$$(\wedge\, m \mid \underset{u}{[\, m \,]}^{u'} \in \mathcal{B}^1 : \{m\} \subseteq s_{(u,u')}) \ .$$

This is exactly hypothesis (4.1).

For the inductive case, we suppose that

$$\underset{u}{(\!|\, \mathcal{B} \,|\!)}^{u'}_{k} \subseteq s_{(u,u')} \tag{4.5}$$

is true for all $k \in \{0, \ldots, n\}$ and for all $(u, u') \in \mathsf{F}^*_{\mathcal{B}}(\{(x, y)\})$ and we show that

$$\underset{u}{(\!|\, \mathcal{B} \,|\!)}^{u'}_{n+1} \subseteq s_{(u,u')}$$

is also true for all $(u, u') \in \mathsf{F}^*_{\mathcal{B}}(\{(x, y)\})$. By definition of $(\!|_u \mathcal{B} |\!)^{u'}_{n+1}$ for $n \in \mathbb{N}$, it suffices to prove that

$$\begin{aligned}
&(\cup\, m, v \mid [_u\, m\, ]^v \in \mathcal{B}^1 : \{m\} \bullet (\!|_v \mathcal{B} |\!)^{u'}_n) \\
&\cup\, (\cup\, c, z, r, w \mid [_u\, c \downarrow_z \uparrow^w r\, ]^{u'} \in \mathcal{B}^2 : \{c\} \bullet (\!|_z \mathcal{B} |\!)^w_n \bullet \{r\}) \\
&\cup\, (\cup\, c, z, r, w, v, n_1, n_2 \mid [_u\, c \downarrow_z \uparrow^w r\, ]^v \in \mathcal{B}^2 \wedge\ n_1, n_2 \in \mathbb{N} \\
&\qquad \wedge\ n_1 + n_2 = n - 1 : \{c\} \bullet (\!|_z \mathcal{B} |\!)^w_{n_1} \bullet \{r\} \bullet (\!|_v \mathcal{B} |\!)^{u'}_{n_2}) \\
&\subseteq s_{(u,u')} \ .
\end{aligned}$$

Note that, by definition of $\mathsf{F}^*_{\mathcal{B}}$ and since $(u, u') \in \mathsf{F}^*_{\mathcal{B}}(\{(x, y)\})$, then each $(v, u')$ and $(z, w)$ in the previous inclusion is in $\mathsf{F}^*_{\mathcal{B}}(\{(x, y)\})$. So, the induction hypotheses are applicable for them. By induction hypothesis (4.5), by transitivity of $\subseteq$ and by monotonicity of $\cup$ and $\bullet$, it suffices to prove that

$$\begin{aligned}
&(\cup\, m, v \mid [_u\, m\, ]^v \in \mathcal{B}^1 : \{m\} \bullet s_{(v,u')}) \\
&\cup\, (\cup\, c, z, r, w \mid [_u\, c \downarrow_z \uparrow^w r\, ]^{u'} \in \mathcal{B}^2 : \{c\} \bullet s_{(z,w)} \bullet \{r\}) \\
&\cup\, (\cup\, c, z, r, w, v, n_1, n_2 \mid [_u\, c \downarrow_z \uparrow^w r\, ]^v \in \mathcal{B}^2 \wedge\ n_1, n_2 \in \mathbb{N} \\
&\qquad \wedge\ n_1 + n_2 = n - 1 : \{c\} \bullet s_{(z,w)} \bullet \{r\} \bullet s_{(v,u')}) \\
&\subseteq s_{(u,u')} \ .
\end{aligned}$$

Since the body of the last quantification does not use the quantified variables $n_1$ and $n_2$, by arithmetics and by set theory, they can be removed. Then, by set theory, it

suffices to prove that

$$(\wedge\, m, v \mid [\underset{u}{\ } m \overset{v}{\ }] \in \mathcal{B}^1 : \{m\} \bullet s_{(v,u')} \subseteq s_{(u,u')}) \ ,$$

$$(\wedge\, c, z, r, w \mid [\underset{u}{\ } c \underset{z}{\downarrow} \overset{w}{\uparrow} r \overset{u'}{\ }] \in \mathcal{B}^2 : \{c\} \bullet s_{(z,w)} \bullet \{r\} \subseteq s_{(u,u')}) \ ,$$

$$(\wedge\, c, z, r, w, v \mid [\underset{u}{\ } c \underset{z}{\downarrow} \overset{w}{\uparrow} r \overset{v}{\ }] \in \mathcal{B}^2 : \{c\} \bullet s_{(z,w)} \bullet \{r\} \bullet s_{(v,u')} \subseteq s_{(u,u')}) \ .$$

These are exactly hypotheses (4.2), (4.3) and (4.4).

**Axiom** (3.16) **is valid:** By definition of the language model, we suppose that for all $u, u', u'' \in V$, $x_2' \in V'$ and $m \in \Sigma_{\mathbf{i}} \cup \{\varepsilon\}$ such that $[_u\, m\, ]^{u'} \in \mathcal{B}^1$,

$$b_{x_2'}^{(u'',u)} \bullet \{m\} \subseteq (\cup\, y_2'' \mid [\underset{x_2'}{\ } m \overset{y_2''}{\ }] \in \mathcal{C}^1 : \{m\} \bullet b_{y_2''}^{(u'',u')}) \ . \tag{4.6}$$

Also, we suppose that for all $u, u', u'', z, w \in V$, $x_2' \in V'$, $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$ such that $[_u\, c \downarrow_z \uparrow^w r\, ]^{u'} \in \mathcal{B}^2$,

$$b_{x_2'}^{(u'',u)} \bullet \{c\} \subseteq (\cup\, z', w', y_2'' \mid [\underset{x_2'}{\ } c \underset{z'}{\downarrow} \overset{w'}{\uparrow} r \overset{y_2''}{\ }] \in \mathcal{C}^2 : b_{x_2'}^{(u'',u)} \bullet \{c\} \bullet b_{z'}^{(z,z)} \bullet b_{z'}^{(z,z)}) \ . \tag{4.7}$$

Moreover, we suppose that for all $u, u', u'', z, w \in V$, $x_2', z', w'' \in V'$, $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$ such that $[_u\, c \downarrow_z \uparrow^w r\, ]^{u'} \in \mathcal{B}^2$,

$$\begin{aligned} &(\cup\, w', y_2'' \mid [\underset{x_2'}{\ } c \underset{z'}{\downarrow} \overset{w'}{\uparrow} r \overset{y_2''}{\ }] \in \mathcal{C}^2 : b_{x_2'}^{(u'',u)} \bullet \{c\} \bullet b_{z'}^{(z,z)} \bullet (\!|\underset{z'}{\ } \mathcal{C} \overset{w''}{\ }|\!) \bullet b_{w''}^{(z,w)} \bullet \{r\}) \\ \subseteq\ &(\cup\, y_2'' \mid [\underset{x_2'}{\ } c \underset{z'}{\downarrow} \overset{w''}{\uparrow} r \overset{y_2''}{\ }] \in \mathcal{C}^2 : \{c\} \bullet (\!|\underset{z'}{\ } \mathcal{C} \overset{w''}{\ }|\!) \bullet \{r\} \bullet b_{y_2''}^{(u'',u')}) \ . \end{aligned} \tag{4.8}$$

We show that

$$b_{x_2}^{(y',x)} \bullet (\!|\underset{x}{\ } \mathcal{B} \overset{y}{\ }|\!) \subseteq (\cup\, y_2' \mid y_2' \in V' : (\!|\underset{x_2}{\ } \mathcal{C} \overset{y_2'}{\ }|\!) \bullet b_{y_2'}^{(y',y)}) \ .$$

By definition of $(\!|_x\, \mathcal{B}\, |\!)^y$, by distributivity of $\bullet$ over $\cup$ and by set theory, it suffices to prove that for all $n \in \mathbb{N}$,

$$b_{x_2}^{(y',x)} \bullet (\!|\underset{x}{\ } \mathcal{B} \overset{y}{\ }|\!)_n \subseteq (\cup\, y_2' \mid y_2' \in V' : (\!|\underset{x_2}{\ } \mathcal{C} \overset{y_2'}{\ }|\!) \bullet b_{y_2'}^{(y',y)}) \ .$$

The proof is done by generalized induction over $n$. For the base case ($n = 0$), we show that

$$b_{x_2}^{(y',x)} \bullet (\!|\underset{x}{\ } \mathcal{B} \overset{y}{\ }|\!)_0 \subseteq (\cup\, y_2' \mid y_2' \in V' : (\!|\underset{x_2}{\ } \mathcal{C} \overset{y_2'}{\ }|\!) \bullet b_{y_2'}^{(y',y)}) \ .$$

By definition of $(\!|_x\, \mathcal{B}\, |\!)^y_0$, distributivity of $\bullet$ over $\cup$ and set theory, it suffices to prove independently that, for all unary blocks $[\!|_x\, m\, |\!]^y \in \mathcal{B}^1$,

$$b^{(y',x)}_{x_2} \bullet \{m\} \subseteq (\cup\, y'_2 \mid y'_2 \in V' : \underset{x_2}{(\!|}\, \mathcal{C}\, \overset{y'_2}{|\!)} \bullet b^{(y',y)}_{y'_2}) \ .$$

We prove it.

$$\begin{array}{ll}
 & b^{(y',x)}_{x_2} \bullet \{m\} \\
\subseteq & \quad \{\!\{ \text{ Hypothesis (4.6) } \}\!\} \\
 & (\cup\, y'_2 \mid [\!|_{x_2}\, m\, |\!]^{y'_2} \in \mathcal{C}^1 : \{m\} \bullet b^{(y',y)}_{y'_2}) \\
\subseteq & \quad \{\!\{ \text{ Since } [\!|_{x_2}\, m\, |\!]^{y'_2} \in \mathcal{C}^1, \text{ axiom (3.2) can be used \& Monotonicity of } \bullet \text{ and } \cup \ \}\!\} \\
 & (\cup\, y'_2 \mid [\!|_{x_2}\, m\, |\!]^{y'_2} \in \mathcal{C}^1 : (\!|_{x_2}\, \mathcal{C}\, |\!)^{y'_2} \bullet b^{(y',y)}_{y'_2}) \\
\subseteq & \quad \{\!\{ \text{ By definition: } [\!|_{x_2}\, m\, |\!]^{y'_2} \in \mathcal{C}^1 \Rightarrow y'_2 \in V' \text{ \& Range weakening } \}\!\} \\
 & (\cup\, y'_2 \mid y'_2 \in V' : (\!|_{x_2}\, \mathcal{C}\, |\!)^{y'_2} \bullet b^{(y',y)}_{y'_2})
\end{array}$$

For the inductive case, we suppose that

$$b^{(y',x)}_{x_2} \bullet \underset{x}{(\!|}\, \mathcal{B}\, \overset{y}{\underset{k}{|\!)}} \subseteq (\cup\, y'_2 \mid y'_2 \in V' : \underset{x_2}{(\!|}\, \mathcal{C}\, \overset{y'_2}{|\!)} \bullet b^{(y',y)}_{y'_2})$$

is true for all $k \in \{0, \ldots, n\}$ and for all $x, y, y' \in V$ and $x_2 \in V'$ and we show that

$$b^{(y',x)}_{x_2} \bullet \underset{x}{(\!|}\, \mathcal{B}\, \overset{y}{\underset{n+1}{|\!)}} \subseteq (\cup\, y'_2 \mid y'_2 \in V' : \underset{x_2}{(\!|}\, \mathcal{C}\, \overset{y'_2}{|\!)} \bullet b^{(y',y)}_{y'_2})$$

is also true. By definition of $(\!|_x\, \mathcal{B}\, |\!)^y_{n+1}$, by distributivity of $\bullet$ over $\cup$ and by set theory, it suffices to show independently that,

- for all $v \in V$ and unary blocks $[\!|_x\, m\, |\!]^v \in \mathcal{B}^1$,

$$b^{(y',x)}_{x_2} \bullet \{m\} \bullet \underset{v}{(\!|}\, \mathcal{B}\, \overset{y}{\underset{n}{|\!)}} \subseteq (\cup\, y'_2 \mid y'_2 \in V' : \underset{x_2}{(\!|}\, \mathcal{C}\, \overset{y'_2}{|\!)} \bullet b^{(y',y)}_{y'_2}) \ , \tag{4.9}$$

- for all binary blocks $[\!|_x\, c \downarrow_z \uparrow^w r\, |\!]^y \in \mathcal{B}^2$,

$$b^{(y',x)}_{x_2} \bullet \{c\} \bullet \underset{z}{(\!|}\, \mathcal{B}\, \overset{w}{\underset{n}{|\!)}} \bullet \{r\} \subseteq (\cup\, y'_2 \mid y'_2 \in V' : \underset{x_2}{(\!|}\, \mathcal{C}\, \overset{y'_2}{|\!)} \bullet b^{(y',y)}_{y'_2}) \ , \tag{4.10}$$

- for all $v \in V$, binary blocks $\lfloor_x\, c {\downarrow_z} {\uparrow^w}\, r \,\rceil^v \in \mathcal{B}^2$ and $n_1, n_2 \in \mathbb{N}$ such that $n_1 + n_2 = n - 1$,

$$b^{(y',x)}_{x_2} \bullet \{c\} \bullet \underset{z}{\langle\!(} \mathcal{B} \underset{n_1}{)\!\rangle}^{w} \bullet \{r\} \bullet \underset{v}{\langle\!(} \mathcal{B} \underset{n_2}{)\!\rangle}^{y} \subseteq (\cup\, y'_2 \mid y'_2 \in V' : \underset{x_2}{\langle\!(} \mathcal{C} )\!\rangle^{y'_2} \bullet b^{(y',y)}_{y'_2}) \ . \tag{4.11}$$

For (4.9),

$$\begin{array}{ll}
& b^{(y',x)}_{x_2} \bullet \{m\} \bullet (\!|_v\, \mathcal{B}\, |\!)^y_n \\
\subseteq & \quad \{\!\{ \text{ Hypothesis (4.6) \& Monotonicity of } \bullet \ \}\!\} \\
& (\cup\, v' \mid \lfloor_{x_2}\, m\, \rceil^{v'} \in \mathcal{C}^1 : \{m\} \bullet b^{(y',v)}_{v'}) \bullet (\!|_v\, \mathcal{B}\, |\!)^y_n \\
= & \quad \{\!\{ \text{ Distributivity of } \bullet \text{ over } \cup \ \}\!\} \\
& (\cup\, v' \mid \lfloor_{x_2}\, m\, \rceil^{v'} \in \mathcal{C}^1 : \{m\} \bullet b^{(y',v)}_{v'} \bullet (\!|_v\, \mathcal{B}\, |\!)^y_n) \\
\subseteq & \quad \{\!\{ \text{ Induction hypothesis \& Monotonicity of } \bullet \text{ and } \cup \ \}\!\} \\
& (\cup\, v' \mid \lfloor_{x_2}\, m\, \rceil^{v'} \in \mathcal{C}^1 : \{m\} \bullet (\cup\, y'_2 \mid y'_2 \in V' : (\!|_{v'}\, \mathcal{C}\, |\!)^{y'_2} \bullet b^{(y',y)}_{y'_2})) \\
= & \quad \{\!\{ \text{ Distributivity of } \bullet \text{ over } \cup \text{ \& Nesting } \}\!\} \\
& (\cup\, y'_2 \mid y'_2 \in V' : (\cup\, v' \mid \lfloor_{x_2}\, m\, \rceil^{v'} \in \mathcal{C}^1 : \{m\} \bullet (\!|_{v'}\, \mathcal{C}\, |\!)^{y'_2}) \bullet b^{(y',y)}_{y'_2}) \\
\subseteq & \quad \{\!\{ \text{ Since } \lfloor_{x_2}\, m\, \rceil^{v'} \in \mathcal{C}^1, \text{ axiom (3.2) can be used \& Axiom (3.4) \& Monotonicity of } \bullet \text{ and } \cup \ \}\!\} \\
& (\cup\, y'_2 \mid y'_2 \in V' : (\cup\, v' \mid \lfloor_{x_2}\, m\, \rceil^{v'} \in \mathcal{C}^1 : (\!|_{x_2}\, \mathcal{C}\, |\!)^{y'_2}) \bullet b^{(y',y)}_{y'_2}) \\
\subseteq & \quad \{\!\{ \text{ Set theory } \}\!\} \\
& (\cup\, y'_2 \mid y'_2 \in V' : (\!|_{x_2}\, \mathcal{C}\, |\!)^{y'_2} \bullet b^{(y',y)}_{y'_2}) \ .
\end{array}$$

For (4.10) and (4.11), we first prove that for all $v, v' \in V$, binary blocks $\lfloor_x\, c {\downarrow_z} {\uparrow^w}\, r\, \rceil^v \in \mathcal{B}^2$ and $k \in \mathbb{N}$ such that $k \leqslant n$,

$$b^{(v',x)}_{x_2} \bullet \{c\} \bullet \underset{z}{\langle\!(} \mathcal{B} \underset{k}{)\!\rangle}^{w} \bullet \{r\} \subseteq (\cup\, v'_2 \mid v'_2 \in V' : \underset{x_2}{\langle\!(} \mathcal{C} )\!\rangle^{v'_2} \bullet b^{(v',v)}_{v'_2}) \ . \tag{4.12}$$

For (4.12),

$$\begin{array}{ll}
& b^{(v',x)}_{x_2} \bullet \{c\} \bullet (\!|_z\, \mathcal{B}\, |\!)^w_k \bullet \{r\} \\
\subseteq & \quad \{\!\{ \text{ Hypothesis (4.7) \& Monotonicity of } \bullet \ \}\!\} \\
& (\cup\, z', w', v'_2 \mid \lfloor_{x_2}\, c {\downarrow_{z'}} {\uparrow^{w'}}\, r\, \rceil^{v'_2} \in \mathcal{C}^2 : b^{(v',x)}_{x_2} \bullet \{c\} \bullet b^{(z,z)}_{z'} \bullet b^{(z,z)}_{z'}) \bullet (\!|_z\, \mathcal{B}\, |\!)^w_k \bullet \{r\} \\
= & \quad \{\!\{ \text{ Distributivity of } \bullet \text{ over } \cup \ \}\!\} \\
& (\cup\, z', w', v'_2 \mid \lfloor_{x_2}\, c {\downarrow_{z'}} {\uparrow^{w'}}\, r\, \rceil^{v'_2} \in \mathcal{C}^2 : b^{(v',x)}_{x_2} \bullet \{c\} \bullet b^{(z,z)}_{z'} \bullet b^{(z,z)}_{z'} \bullet (\!|_z\, \mathcal{B}\, |\!)^w_k \bullet \{r\}) \\
\subseteq & \quad \{\!\{ \text{ Induction hypothesis \& Monotonicity of } \bullet \text{ and } \cup \ \}\!\}
\end{array}$$

$$
\begin{array}{ll}
 & (\cup\ z', w', v_2' \mid \lfloor_{x_2}\, c{\downarrow_{z'}}{\uparrow^{w'}}\, r\,\rfloor^{v_2'} \in \mathcal{C}^2 : b_{x_2}^{(v',x)} \bullet \{c\} \bullet b_{z'}^{(z,z)} \bullet (\cup\ w'' \mid w'' \in V' : \\
 & (\!|_{z'}\, \mathcal{C}\, |\!)^{w''} \bullet b_{w''}^{(z,w)}) \bullet \{r\}) \\
= & \quad \{\!\{ \text{ Distributivity of } \bullet \text{ over } \cup \text{ \& Nesting } \}\!\} \\
 & (\cup\ z', w'' \mid z' \in V' \wedge w'' \in V' : (\cup\ w', v_2' \mid \lfloor_{x_2}\, c{\downarrow_{z'}}{\uparrow^{w'}}\, r\,\rfloor^{v_2'} \in \mathcal{C}^2 : b_{x_2}^{(v',x)} \bullet \{c\} \bullet b_{z'}^{(z,z)} \bullet \\
 & (\!|_{z'}\, \mathcal{C}\, |\!)^{w''} \bullet b_{w''}^{(z,w)} \bullet \{r\})) \\
\subseteq & \quad \{\!\{ \text{ Hypothesis (4.8) \& Monotonicity of } \cup \text{ }\}\!\} \\
 & (\cup\ z', w'' \mid z' \in V' \wedge w'' \in V' : (\cup\ v_2' \mid \lfloor_{x_2}\, c{\downarrow_{z'}}{\uparrow^{w''}}\, r\,\rfloor^{v_2'} \in \mathcal{C}^2 : \{c\} \bullet (\!|_{z'}\, \mathcal{C}\, |\!)^{w''} \bullet \{r\} \bullet \\
 & b_{v_2'}^{(v',v)})) \\
= & \quad \{\!\{ \text{ Nesting \& Distributivity of } \bullet \text{ over } \cup \text{ }\}\!\} \\
 & (\cup\ v_2' \mid v_2' \in V' : (\cup\ z', w'' \mid \lfloor_{x_2}\, c{\downarrow_{z'}}{\uparrow^{w''}}\, r\,\rfloor^{v_2'} \in \mathcal{C}^2 : \{c\} \bullet (\!|_{z'}\, \mathcal{C}\, |\!)^{w''} \bullet \{r\}) \bullet b_{v_2'}^{(v',v)}) \\
\subseteq & \quad \{\!\{ \text{ Since } \lfloor_{x_2}\, c{\downarrow_{z'}}{\uparrow^{w''}}\, r\,\rfloor^{v_2'} \in \mathcal{C}^2, \text{ axiom (3.3) can be used \& Monotonicity of } \bullet \\
 & \quad \text{and } \cup \text{ }\}\!\} \\
 & (\cup\ v_2' \mid v_2' \in V' : (\cup\ z', w'' \mid \lfloor_{x_2}\, c{\downarrow_{z'}}{\uparrow^{w''}}\, r\,\rfloor^{v_2'} \in \mathcal{C}^2 : (\!|_{x_2}\, \mathcal{C}\, |\!)^{v_2'}) \bullet b_{v_2'}^{(v',v)}) \\
\subseteq & \quad \{\!\{ \text{ Set theory } \}\!\} \\
 & (\cup\ v_2' \mid v_2' \in V' : (\!|_{x_2}\, \mathcal{C}\, |\!)^{v_2'} \bullet b_{v_2'}^{(v',v)})\ .
\end{array}
$$

Now, the proof of (4.10) is direct with (4.12) and appropriate substitution.

For (4.11),

$$
\begin{array}{ll}
 & b_{x_2}^{(y',x)} \bullet \{c\} \bullet (\!|_z\, \mathcal{B}\, |\!)_{n_1}^{w} \bullet \{r\} \bullet (\!|_v\, \mathcal{B}\, |\!)_{n_2}^{y} \\
\subseteq & \quad \{\!\{ \text{ Inclusion (4.12) \& Monotonicity of } \bullet \text{ }\}\!\} \\
 & (\cup\ v_2' \mid v_2' \in V' : (\!|_{x_2}\, \mathcal{C}\, |\!)^{v_2'} \bullet b_{v_2'}^{(y',v)}) \bullet (\!|_v\, \mathcal{B}\, |\!)_{n_2}^{y} \\
= & \quad \{\!\{ \text{ Distributivity of } \bullet \text{ over } \cup \text{ }\}\!\} \\
 & (\cup\ v_2' \mid v_2' \in V' : (\!|_{x_2}\, \mathcal{C}\, |\!)^{v_2'} \bullet b_{v_2'}^{(y',v)} \bullet (\!|_v\, \mathcal{B}\, |\!)_{n_2}^{y}) \\
\subseteq & \quad \{\!\{ \text{ Induction hypothesis \& Monotonicity of } \bullet \text{ and } \cup \text{ }\}\!\} \\
 & (\cup\ v_2' \mid v_2' \in V' : (\!|_{x_2}\, \mathcal{C}\, |\!)^{v_2'} \bullet (\cup\ y_2' \mid y_2' \in V' : (\!|_{v_2'}\, \mathcal{C}\, |\!)^{y_2'} \bullet b_{y_2'}^{(y',y)})) \\
= & \quad \{\!\{ \text{ Distributivity of } \bullet \text{ over } \cup \text{ \& Nesting } \}\!\} \\
 & (\cup\ y_2' \mid y_2' \in V' : (\cup\ v_2' \mid v_2' \in V' : (\!|_{x_2}\, \mathcal{C}\, |\!)^{v_2'} \bullet\ (\!|_{v_2'}\, \mathcal{C}\, |\!)^{y_2'}) \bullet b_{y_2'}^{(y',y)}) \\
\subseteq & \quad \{\!\{ \text{ Axiom (3.4) \& Monotonicity of } \bullet \text{ and } \cup \text{ }\}\!\} \\
 & (\cup\ y_2' \mid y_2' \in V' : (\cup\ v_2' \mid v_2' \in V' : (\!|_{x_2}\, \mathcal{C}\, |\!)^{y_2'}) \bullet b_{y_2'}^{(y',y)}) \\
\subseteq & \quad \{\!\{ \text{ Set theory \& Monotonicity of } \bullet \text{ and } \cup \text{ }\}\!\} \\
 & (\cup\ y_2' \mid y_2' \in V' : (\!|_{x_2}\, \mathcal{C}\, |\!)^{y_2'} \bullet b_{y_2'}^{(y',y)})\ .
\end{array}
$$

■

## 4.2 Completeness of the Equational Theory of VPKA over the Language Model under Interpretation $\mathcal{L}$

We show the following theorem.

**Theorem 4.2** (Completeness of the equational theory of VPKA over the language model under interpretation $\mathcal{L}$)**.** *Let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be three disjoint finite sets such that at least one of them is nonempty. Let $p$ and $q$ be visibly pushdown regular expressions over the same tripartition denoting the same visibly pushdown language. Then, the axiomatic system of visibly pushdown Kleene algebra can derive $p = q$. In other words, for any* VPRE*s $p$ and $q$,*

$$\mathcal{L}(p) = \mathcal{L}(q) \implies \vdash p = q \ .$$

The proof of Theorem 4.2 is very involved. We use an approach similar to Conway [16] and Kozen [25], but we also use some ideas of Wagner [51]. The key step of the proof is to encode (semi-)visibly pushdown automata directly in visibly pushdown Kleene algebra. This allows us to manipulate (S-)VPA in a purely algebraic setting. The algebraic encoding is defined using matrices over a Kleene algebra like Kozen's finite automata encoding [25], but the encoding is different. Recall that, by definition, any visibly pushdown Kleene algebra is a Kleene algebra, and the family of matrices over a Kleene algebra again forms a Kleene algebra (see Section 2.1). So visibly pushdown regular expressions can be used as entries of matrices. Also, matrices using these expressions can be manipulated with the standard operators of Kleene algebra.

At first glance, the algebraic encoding of (semi-)visibly pushdown automata that can lead to the definition of a halting algorithm (to ensure decidability) does not seem easy. Known algebraic encodings of pushdown automata are of no help here because they usually give matrices of infinite size like in [35]. So, the particular structure of (S-)VPA must be exploited to define such an encoding.

**Definition 4.3** (Algebraic encoding of a semi-visibly pushdown automaton)**.** Let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be three disjoint finite sets such that at least one of them is nonempty. Let $S := \{s_1, s_2, \ldots, s_{|S|}\}$ be a nonempty finite set. Let

$$\mathcal{A} := (S, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma \cup \{\bot\}, \delta, I, F)$$

be a semi-visibly pushdown automaton. The *algebraic encoding of* $\mathcal{A}$ is the structure

$$(\overrightarrow{I}, \mathbf{WM}, \mathbf{T_c}, \mathbf{T}_\bot, \varepsilon_\bot, \varepsilon_{\not\bot}, \overrightarrow{F})$$

in which:

- $\overrightarrow{I}$ and $\overrightarrow{F}$ are "Boolean" column vectors[1] of size $|S|$ representing the initial states and the accepting states;

- each entry $i, j$ of the square matrix $\mathbf{T_c}$ (of size $|S| \times |S|$) is defined by

$$\mathbf{T_c}[i,j] := (\sum\, c \mid c \in \Sigma_{\mathbf{c}} \wedge (\exists\, d \mid d \in \Gamma : (s_i, c, \lambda; s_j, d) \in \delta) : c) \ .$$

  This matrix represents the possibly pending calls;

- each entry $i, j$ of the square matrix $\mathbf{T}_\perp$ (of size $|S| \times |S|$) is defined by

$$\mathbf{T}_\perp[i,j] := (\sum\, r \mid r \in \Sigma_{\mathbf{r}} \wedge (s_i, r, \perp; s_j, \perp) \in \delta : r) \ .$$

  This matrix represents the possibly pending returns;

- each entry $i, j$ of the square matrix $\varepsilon_\perp$ (of size $|S| \times |S|$) is defined by

$$\varepsilon_\perp[i,j] := \begin{cases} 1 & \text{if } (s_i, \varepsilon, \perp; s_j, \perp) \in \delta, \\ 0 & \text{otherwise.} \end{cases}$$

  This matrix represents the $\varepsilon$-transitions that test if the top symbol on the stack is the bottom-of-stack symbol;

- each entry $i, j$ of the square matrix $\varepsilon_{\not\perp}$ (of size $|S| \times |S|$) is defined by

$$\varepsilon_{\not\perp}[i,j] := \begin{cases} 1 & \text{if for all } d \in \Gamma, (s_i, \varepsilon, d; s_j, \perp) \in \delta, \\ 0 & \text{otherwise.} \end{cases}$$

  This matrix represents the $\varepsilon$-transitions that seal the current stack by a $\perp$ symbol;

- each entry $i, j$ of the square matrix $\mathbf{WM}$ (of size $|S| \times |S|$) represents the well-matched words that can be generated from a state $s_i$ of the automaton to a state $s_j$ without using an $\varepsilon$-transition, and it is defined by

$$\mathbf{WM}[i,j] := \underset{s_i}{\overset{s_j}{(\!|\, \mathcal{B} \,|\!)}}$$

  where the list of blocks $\mathcal{B}$ (identical for all entries of the matrix) represents the encoding of the structure of the semi-visibly pushdown automaton (except for $\varepsilon$-transitions) using the following rules:

  A. The set of labels $S$ is used.

[1] Boolean column vectors in this case are vectors that contain only 0s and 1s, where 0 and 1 are the constants of VPREs.

B. For all states $s_i \in S$,

$$\left[\begin{smallmatrix} s_i \\ \\ s_i \end{smallmatrix}\right. 1 \left.\begin{smallmatrix} s_i \\ \\ \end{smallmatrix}\right] \in \mathcal{B} \ .$$

C. For all internal actions $a \in \Sigma_{\mathbf{i}}$ and states $s_i, s_j \in S$,

$$\left[\begin{smallmatrix} s_i \\ \\ \end{smallmatrix}\right. a \left.\begin{smallmatrix} s_j \\ \\ \end{smallmatrix}\right] \in \mathcal{B} \Leftrightarrow (s_i, a, \lambda; s_j, \lambda) \in \delta \ .$$

D. For all call actions $c \in \Sigma_{\mathbf{c}}$, return actions $r \in \Sigma_{\mathbf{r}}$ and states $s_i, s_j, s_k, s_l \in S$,

$$\left[\begin{smallmatrix} s_i & & s_j \\ & \downarrow \ \uparrow & \\ s_k & & \end{smallmatrix}\right. c \quad r \left.\begin{smallmatrix} \\ \\ s_l \end{smallmatrix}\right] \in \mathcal{B} \Leftrightarrow (\exists\ d \mid d \in \Gamma : (s_i, c, \lambda; s_k, d) \in \delta \wedge (s_j, r, d; s_l, \lambda) \in \ \delta) \ .$$

The *language accepted by* $\mathcal{A}$ is denoted by (see explanation below)

$$\overrightarrow{I}^{\,t} \bullet (\mathbf{T}_\perp + \mathbf{WM} + \varepsilon_\perp + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \ . \tag{4.13}$$

Note that, for matrices of size $1 \times 1$, we use the matrix $\left[\begin{smallmatrix} x \end{smallmatrix}\right]$ as a VPRE $x$. ♦

*Remark* 4.4 (Algebraic encoding of a visibly pushdown automaton). Let $\mathcal{A}$ be a visibly pushdown automaton. Recall that a visibly pushdown automaton is a semi-visibly pushdown automaton that does not have $\varepsilon$-transitions. So, by Definition 4.3, the algebraic encoding of $\mathcal{A}$ is a structure

$$(\overrightarrow{I}, \mathbf{WM}, \mathbf{T_c}, \mathbf{T}_\perp, \mathbf{0}, \mathbf{0}, \overrightarrow{F}) \ .$$

When dealing explicitly with visibly pushdown automata, the abbreviation

$$(\overrightarrow{I}, \mathbf{WM}, \mathbf{T_c}, \mathbf{T}_\perp, \overrightarrow{F})$$

for the algebraic encoding of $\mathcal{A}$ is used and the language accepted by $\mathcal{A}$ is denoted by

$$\overrightarrow{I}^{\,t} \bullet (\mathbf{T}_\perp + \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \ . \qquad ♦$$

The algebraic encoding is mainly concerned about getting rid of the explicit stack in the automaton. Note that the exact stack symbol pushed on the stack when reading a call action is important only if this symbol is used to read a return action. So, this situation occurs only if an affix of the current word is well-matched and contains these call and return actions. So, this situation can be taken care of without using an explicit stack by using the matrix $\mathbf{WM}$ which allows a binary block only if the call action and the return action use the same stack symbol. In any other cases, it is not useful to know the exact stack symbol used, but it is essential to know whether the stack is empty or not. So, the encoding forgets the stack symbol for matrices $\mathbf{T_c}$ and $\varepsilon_{\not\perp}$.

Assume that $\mathbf{WM}[i,j]$ gives the set of well-matched words that can be read by the S-VPA without using $\varepsilon$-transitions. This assumption is easy to accept because the expression $\mathbf{WM}[i,j]$ encodes the structure of the automaton. So, the language (4.13) can be understood in the following way:

- First, every accepted word of the automaton starts in an initial state (a state of $\overrightarrow{I}$) with an empty stack.
- The expression $(\mathbf{T}_\perp + \mathbf{WM} + \varepsilon_\perp + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star$ represents the possibilities that the automaton has to accept the word *without leaving an empty stack (or a stack sealed with a $\perp$ symbol)*:
  - it can read a return action (transition) of $\mathbf{T}_\perp$ (so, the stack remains the same);
  - it can read a well-matched word by using $\mathbf{WM}$ (so, the stack remains the same);
  - it can follow an $\varepsilon$-transition of $\varepsilon_\perp$ (so, the stack remains the same);
  - it can read any finite nonzero number of calls ($\mathbf{T_c}$) followed by any well-matched words of $\mathbf{WM}$, only if this is followed immediately by an $\varepsilon$-transition that seals the stack with a $\perp$ symbol (so, the stack grows but is nevertheless sealed with a $\perp$ symbol). Note that, after reading a call action of $\mathbf{T_c}$, the stack is not empty and has a symbol of $\Gamma$ on top of it, so, the transitions of $\mathbf{T}_\perp$ and $\varepsilon_\perp$ cannot be applied before following a transition of $\varepsilon_{\not\perp}$.
- The expression $(\mathbf{T_c} \bullet \mathbf{WM})^\star$ represents the possibilities that the automaton has to accept a word when processing the first pending call that is not sealed by a $\perp$ symbol: reading a pending call action followed by well-matched words and more pending calls.
- The automaton must stop in a final state (a state of $\overrightarrow{F}$).

**Lemma 4.5** (Properties of $\mathbf{WM}$)**.** *Let $\Sigma_\mathbf{i}$, $\Sigma_\mathbf{c}$ and $\Sigma_\mathbf{r}$ be three disjoint finite sets such that at least one of them is nonempty. Let $S := \{s_1, s_2, \ldots, s_{|S|}\}$ be a nonempty finite set. Let $\mathcal{A} := (S, \Sigma_\mathbf{i}, \Sigma_\mathbf{c}, \Sigma_\mathbf{r}, \Gamma \cup \{\perp\}, \delta, I, F)$ be a semi-visibly pushdown automaton. Let $\mathbf{WM}$ be the matrix of the algebraic encoding of $\mathcal{A}$ described in Definition 4.3. Then,*

*(i)* $\mathbf{I} \leqslant \mathbf{WM}$;

*(ii)* $\mathbf{WM} \bullet \mathbf{WM} = \mathbf{WM}$;

*(iii)* $\mathbf{WM}^\star = \mathbf{WM}$.

*Proof.* We first show property i. By definitions of $\mathbf{I}$ and $\mathbf{WM}$, it suffices to show that for every $i, j \in \{1, 2, \ldots, |S|\}$ such that $i \neq j$, $0 \leqslant (\!|_{s_i} \mathcal{B} |\!)^{s_j}$ and for every $i \in \{1, 2, \ldots, |S|\}$, $1 \leqslant (\!|_{s_i} \mathcal{B} |\!)^{s_i}$. The first inequation is trivial by the fact that 0 is the minimum element of the algebra (simple Kleene algebraic reasoning). For the second inequation, by condition B of the definition of $\mathcal{B}$, there is a unary block $[_{s_i} 1 ]^{s_i}$ in $\mathcal{B}$. So, $(\!|_{s_i} \mathcal{B} |\!)^{s_i} \geqslant 1$ by axiom (3.2).

We now show ii. The case $\geqslant$ is easy by using property i and simple Kleene algebraic reasoning:

$$\mathbf{WM} \bullet \mathbf{WM} \geqslant \mathbf{I} \bullet \mathbf{WM} = \mathbf{WM} \ .$$

The case $\leqslant$ is also easy. By definition of $\bullet$ and $\mathbf{WM}$, it suffices to show that for every $i, j \in \{1, 2, \ldots, |S|\}$,

$$(\sum k \mid 1 \leqslant k \leqslant |S| : (\!|_{s_i} \mathcal{B} |\!)^{s_k} \cdot (\!|_{s_k} \mathcal{B} |\!)^{s_j}) \leqslant (\!|_{s_i} \mathcal{B} |\!)^{s_j} \ .$$

This is trivial by (3.4), monotonicity of $+$ and idempotency of $+$.

We now show iii. The case $\leqslant$ is easy:

$$\begin{array}{ll} & \mathbf{WM}^\star \leqslant \mathbf{WM} \\ \leftarrow & \qquad \{\!\{ \ ^\star\text{-induction axiom} \ \}\!\} \\ & \mathbf{I} + \mathbf{WM} \bullet \mathbf{WM} \leqslant \mathbf{WM} \\ \leftrightarrow & \qquad \{\!\{ \text{ Kleene algebra: } p + q \leqslant r \leftrightarrow p \leqslant r \wedge q \leqslant r \ \}\!\} \\ & \mathbf{I} \leqslant \mathbf{WM} \ \wedge \ \mathbf{WM} \bullet \mathbf{WM} \leqslant \mathbf{WM} \\ & \qquad\qquad \text{— ( Lemma 4.5, properties i and ii . )} \end{array}$$

The case $\geqslant$ is trivial by the $^\star$-unfold axiom and simple Kleene algebraic reasoning. ■

The main idea of the proof of Theorem 4.2 is to use the algebraic encoding of a (semi-)visibly pushdown automaton to derive some results of the theory of (S-)VPA but using only the axiomatic system presented in Section 3.2. The results that are derived here are

- an extension of Kleene's representation theorem for semi-visibly pushdown automata;
- an $\varepsilon$-transition elimination for semi-visibly pushdown automata (that gives visibly pushdown automata);

- the determinization of visibly pushdown automata;
- the synchronization of two deterministic visibly pushdown automata.

Note that the synchronization of two visibly pushdown automata is used instead of the minimization of these automata (unlike [25]). Minimization cannot be used easily here because minimization does not work well for every visibly pushdown automaton; such minimization is not necessarily unique [14] which is a crucial fact for Kozen's proof [25]. To circumvent the problem, we use synchronization between automata which was used by Wagner in [51]. To our knowledge, Wagner was the first to use this idea in an algebraic setting.

Here are the results of the theory of (semi-)visibly pushdown automata that are used in the proof of Theorem 4.2. For each of these results, we give the complete construction, but we only give an overview of each proof since they are lengthy. See Appendices B to E for the complete proofs. We write $\vdash \phi$ to mean that a formula $\phi$ is derivable using the deductive system of visibly pushdown Kleene algebra.

**Theorem 4.6** (Extension of Kleene's representation theorem for semi-visibly pushdown automata)**.** *Let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be three disjoint finite sets such that at least one of them is nonempty. Let $p$ be a visibly pushdown regular expression on $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$. Then, there exists a semi-visibly pushdown automaton $\mathcal{A} := (S, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma \cup \{\bot\}, \delta, I, F)$ that accepts the language $\mathcal{L}(p)$ and the algebraic encoding of $\mathcal{A}$ by the structure $(\overrightarrow{I}, \mathbf{WM}, \mathbf{T_c}, \mathbf{T}_\bot, \varepsilon_\bot, \varepsilon_{\not\bot}, \overrightarrow{F})$ is such that*

$$\vdash p \quad = \quad \overrightarrow{I}^t \bullet (\mathbf{T}_\bot + \mathbf{WM} + \varepsilon_\bot + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\bot})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \ . \qquad (4.14)$$

*Proof sketch (for the complete proof, see Appendix B).* The proof of Theorem 4.6 is done by induction on the structure of $p$. Note that the base cases construct VPAs (which are a special case of S-VPAs) and follow Remark 4.4. Note also that, in our equations between matrices, we sometimes take the liberty of having different entry notation for the matrices. However, the correspondence between the notations will always be clear from the context and, of course, the matrices will always be of the same size.

For the base case of the constant 0, it suffices to construct a one-state VPA without an accepting state. Since there is no accepting state, $\overrightarrow{F}$ is the zero vector and thus

$$\vdash 0 \quad = \quad \overrightarrow{I}^t \bullet (\mathbf{T}_\bot + \mathbf{WM} + \varepsilon_\bot + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\bot})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \begin{bmatrix} 0 \end{bmatrix} \ ,$$

by Kleene algebra.

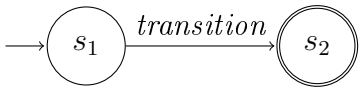
For the base case of the constant 1, it suffices to construct a one-state VPA without transitions and such that its state is accepting. The algebraic encoding of this automaton is

$$\left(\left[\begin{array}{c} 1 \end{array}\right], \left[\begin{array}{c} \langle\!|_{s_1} \, [_{s_1} \, 1 \, ]^{s_1} \, |\!\rangle^{s_1} \end{array}\right], \mathbf{0}, \mathbf{0}, \left[\begin{array}{c} 1 \end{array}\right]\right) \quad .$$

It is easy to see that $\vdash 1 = \langle\!|_{s_1} \, [_{s_1} \, 1 \, ]^{s_1} \, |\!\rangle^{s_1}$. The case $\leqslant$ is direct by (3.2). The other case follows from (3.14) with $s_{(s_1,s_1)} := 1$ and simple Kleene algebraic reasoning. Thus, it is easy to prove (4.14):

$$\begin{array}{ll} & \left[\begin{array}{c} 1 \end{array}\right] \bullet \left(\mathbf{0} + \left[\begin{array}{c} \langle\!|_{s_1} \, [_{s_1} \, 1 \, ]^{s_1} \, |\!\rangle^{s_1} \end{array}\right]\right)^{\star} \bullet \left(\mathbf{0} \bullet \left[\begin{array}{c} \langle\!|_{s_1} \, [_{s_1} \, 1 \, ]^{s_1} \, |\!\rangle^{s_1} \end{array}\right]\right)^{\star} \bullet \left[\begin{array}{c} 1 \end{array}\right] \\ = & \qquad \{\!\{ \text{ Zero of } \cdot \text{ \& Kleene algebra: } 0^* = 1 \text{ \& Identity of } \cdot \text{ and } + \}\!\} \\ & \left[\begin{array}{c} \langle\!|_{s_1} \, [_{s_1} \, 1 \, ]^{s_1} \, |\!\rangle^{s_1} \end{array}\right]^{\star} \\ = & \qquad \{\!\{ \text{ Lemma 4.5, property iii \& Previous result: } \vdash 1 = \langle\!|_{s_1} \, [_{s_1} \, 1 \, ]^{s_1} \, |\!\rangle^{s_1} \}\!\} \\ & 1 \ . \end{array}$$

For the base cases of an internal action $a \in \Sigma_{\mathbf{i}}$, a call action $c \in \Sigma_{\mathbf{c}}$ and a return action $r \in \Sigma_{\mathbf{r}}$, the proofs are similar. It suffices to construct a VPA like this one:




where *transition* represents either the transition

- $(s_1, a, \lambda; s_2, \lambda)$;
- $(s_1, c, \lambda; s_2, d)$ for a $d \in \Gamma$;
- $(s_1, r, \bot; s_2, \bot)$.

Let us prove only the case for an internal action $a \in \Sigma_{\mathbf{i}}$. Let

$$\mathcal{B} := \underset{s_1}{\left[\begin{array}{c} 1 \end{array}\right]}^{s_1}, \ \underset{s_2}{\left[\begin{array}{c} 1 \end{array}\right]}^{s_2}, \ \underset{s_1}{\left[\begin{array}{c} a \end{array}\right]}^{s_2} \quad .$$

The algebraic encoding of this automaton is

$$\left(\left[\begin{array}{c} 1 \\ 0 \end{array}\right], \left[\begin{array}{cc} \langle\!|_{s_1} \, \mathcal{B} \, |\!\rangle^{s_1} & \langle\!|_{s_1} \, \mathcal{B} \, |\!\rangle^{s_2} \\ \langle\!|_{s_2} \, \mathcal{B} \, |\!\rangle^{s_1} & \langle\!|_{s_2} \, \mathcal{B} \, |\!\rangle^{s_2} \end{array}\right], \mathbf{0}, \mathbf{0}, \left[\begin{array}{c} 0 \\ 1 \end{array}\right]\right) \quad .$$

We first prove that $\vdash a = \langle\!|_{s_1} \, \mathcal{B} \, |\!\rangle^{s_2}$. The case $\leqslant$ is direct by (3.2). For the case $\geqslant$,

$$
\begin{array}{ll}
 & (\!|_{s_1} \mathcal{B} |\!)^{s_2} \leqslant a \\
\leftarrow & \qquad \{\!\{ \text{ Axiom (3.14) with } s_{(s_1,s_1)} := 1,\ s_{(s_1,s_2)} := a \text{ and } s_{(s_2,s_2)} := 1 \ \}\!\} \\
 & 1 \leqslant 1 \quad \wedge \quad 1 \cdot 1 \leqslant 1 \quad \wedge \quad a \leqslant a \quad \wedge \quad 1 \cdot a \leqslant a \quad \wedge \quad a \cdot 1 \leqslant a \\
 & \qquad\qquad \text{— ( Identity of } \cdot \text{ \& Reflexivity of } \leqslant \text{ )}
\end{array}
$$

It is easy to prove (4.14):

$$
\begin{array}{ll}
 & \begin{bmatrix} 1 & 0 \end{bmatrix} \bullet \left( \mathbf{0} + \begin{bmatrix} (\!|_{s_1} \mathcal{B} |\!)^{s_1} & (\!|_{s_1} \mathcal{B} |\!)^{s_2} \\ (\!|_{s_2} \mathcal{B} |\!)^{s_1} & (\!|_{s_2} \mathcal{B} |\!)^{s_2} \end{bmatrix} \right)^{\star} \bullet \left( \mathbf{0} \bullet \begin{bmatrix} (\!|_{s_1} \mathcal{B} |\!)^{s_1} & (\!|_{s_1} \mathcal{B} |\!)^{s_2} \\ (\!|_{s_2} \mathcal{B} |\!)^{s_1} & (\!|_{s_2} \mathcal{B} |\!)^{s_2} \end{bmatrix} \right)^{\star} \bullet \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\
= & \qquad \{\!\{ \text{ Zero of } \cdot \text{ \& Kleene algebra: } 0^* = 1 \text{ \& Identity of } \cdot \text{ and } + \ \}\!\} \\
 & \begin{bmatrix} 1 & 0 \end{bmatrix} \bullet \begin{bmatrix} (\!|_{s_1} \mathcal{B} |\!)^{s_1} & (\!|_{s_1} \mathcal{B} |\!)^{s_2} \\ (\!|_{s_2} \mathcal{B} |\!)^{s_1} & (\!|_{s_2} \mathcal{B} |\!)^{s_2} \end{bmatrix}^{\star} \bullet \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\
= & \qquad \{\!\{ \text{ Lemma 4.5, property iii \& Definition of } \bullet \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } + \ \}\!\} \\
 & (\!|_{s_1} \mathcal{B} |\!)^{s_2} \\
= & \qquad \{\!\{ \text{ Previous result: } \vdash a = (\!|_{s_1} \mathcal{B} |\!)^{s_2} \ \}\!\} \\
 & a \ .
\end{array}
$$

For the inductive case $+$, let $p := q_1 + q_2$ where $q_1$ and $q_2$ are VPREs. Suppose there exists a semi-visibly pushdown automaton $\mathcal{A}_1 := (S_1, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_1 \cup \{\bot\}, \delta_1, I_1, F_1)$ that accepts the language $\mathcal{L}(q_1)$ and the algebraic encoding of $\mathcal{A}_1$ by the structure $(\overrightarrow{I_1}, \mathbf{WM}_1, \mathbf{T}_{\mathsf{c}_1}, \mathbf{T}_{\bot_1}, \varepsilon_{\bot_1}, \varepsilon_{\not\bot_1}, \overrightarrow{F_1})$ is such that (4.14) is valid. Also, suppose there exists a semi-visibly pushdown automaton $\mathcal{A}_2 := (S_2, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_2 \cup \{\bot\}, \delta_2, I_2, F_2)$ that accepts the language $\mathcal{L}(q_2)$ and the algebraic encoding of $\mathcal{A}_2$ by the structure $(\overrightarrow{I_2}, \mathbf{WM}_2, \mathbf{T}_{\mathsf{c}_2}, \mathbf{T}_{\bot_2}, \varepsilon_{\bot_2}, \varepsilon_{\not\bot_2}, \overrightarrow{F_2})$ is such that (4.14) is valid.

Without loss of generality, suppose $S_1 \cap S_2 = \emptyset$ and $\Gamma_1 \cap \Gamma_2 = \emptyset$. Define the semi-visibly pushdown automaton

$$\mathcal{A} := (S_1 \cup S_2, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_1 \cup \Gamma_2 \cup \{\bot\}, \delta, I_1 \cup I_2, F_1 \cup F_2)$$

where $\delta := \delta_1 \cup \delta_2 \cup \mathsf{Waste}$, and

$$
\begin{array}{rl}
\mathsf{Waste} := & \{(s, \varepsilon, d; s', \bot) \mid d \in \Gamma_2 \wedge (\exists\ d' \mid d' \in \Gamma_1 : (s, \varepsilon, d'; s', \bot) \in \delta_1)\} \\
 & \cup \{(s, \varepsilon, d; s', \bot) \mid d \in \Gamma_1 \wedge (\exists\ d' \mid d' \in \Gamma_2 : (s, \varepsilon, d'; s', \bot) \in \delta_2)\} \ .
\end{array}
$$

It is easy to see that $\mathcal{A}$ accepts the language $\mathcal{L}(q_1 + q_2)$. Note that the elements of Waste are added just to satisfy the property (2.3) of the definition of S-VPA.

Let $\mathcal{B} := \mathcal{B}_1, \mathcal{B}_2$ (this list of block is used since $S_1 \cap S_2 = \emptyset$ and $\Gamma_1 \cap \Gamma_2 = \emptyset$ by hypothesis, and the construction just adds $\varepsilon$-transitions). The algebraic encoding of this automaton is

$$\left( \overrightarrow{I}, \mathbf{WM}, \left[\begin{array}{c|c} \mathbf{T}_{\mathsf{c}_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}_2} \end{array}\right], \left[\begin{array}{c|c} \mathbf{T}_{\perp_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{\perp_2} \end{array}\right], \left[\begin{array}{c|c} \varepsilon_{\perp_1} & \mathbf{0} \\ \hline \mathbf{0} & \varepsilon_{\perp_2} \end{array}\right], \left[\begin{array}{c|c} \varepsilon_{\not\perp_1} & \mathbf{0} \\ \hline \mathbf{0} & \varepsilon_{\not\perp_2} \end{array}\right], \overrightarrow{F} \right) \quad ,$$

where

$$\overrightarrow{I} := \left[\begin{array}{c} \overrightarrow{I_1} \\ \hline \overrightarrow{I_2} \end{array}\right] \qquad \text{and} \qquad \overrightarrow{F} := \left[\begin{array}{c} \overrightarrow{F_1} \\ \hline \overrightarrow{F_2} \end{array}\right] \quad .$$

First, it is easy to see (but lengthy to prove) that

$$\vdash \mathbf{WM} = \left[\begin{array}{c|c} \mathbf{WM}_1 & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM}_2 \end{array}\right] \quad . \tag{4.15}$$

We now prove (4.14).

$$\begin{aligned}
& \left[\begin{array}{c|c} \overrightarrow{I_1}^t & \overrightarrow{I_2}^t \end{array}\right] \bullet \left( \left[\begin{array}{c|c} \mathbf{T}_{\perp_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{\perp_2} \end{array}\right] + \mathbf{WM} + \left[\begin{array}{c|c} \varepsilon_{\perp_1} & \mathbf{0} \\ \hline \mathbf{0} & \varepsilon_{\perp_2} \end{array}\right] \right. \\
& \qquad \left. + \left( \left( \left[\begin{array}{c|c} \mathbf{T}_{\mathsf{c}_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}_2} \end{array}\right] \bullet \mathbf{WM} \right)^+ \bullet \left[\begin{array}{c|c} \varepsilon_{\not\perp_1} & \mathbf{0} \\ \hline \mathbf{0} & \varepsilon_{\not\perp_2} \end{array}\right] \right)^\star \right) \\
& \bullet \left( \left[\begin{array}{c|c} \mathbf{T}_{\mathsf{c}_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}_2} \end{array}\right] \bullet \mathbf{WM} \right)^\star \bullet \left[\begin{array}{c} \overrightarrow{F_1} \\ \hline \overrightarrow{F_2} \end{array}\right] \\
= & \qquad \{\!\!\{ \text{ Equation (4.15) } \}\!\!\} \\
& \left[\begin{array}{c|c} \overrightarrow{I_1}^t & \overrightarrow{I_2}^t \end{array}\right] \bullet \left( \left[\begin{array}{c|c} \mathbf{T}_{\perp_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{\perp_2} \end{array}\right] + \left[\begin{array}{c|c} \mathbf{WM}_1 & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM}_2 \end{array}\right] + \left[\begin{array}{c|c} \varepsilon_{\perp_1} & \mathbf{0} \\ \hline \mathbf{0} & \varepsilon_{\perp_2} \end{array}\right] \right. \\
& \qquad \left. + \left( \left( \left[\begin{array}{c|c} \mathbf{T}_{\mathsf{c}_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}_2} \end{array}\right] \bullet \left[\begin{array}{c|c} \mathbf{WM}_1 & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM}_2 \end{array}\right] \right)^+ \bullet \left[\begin{array}{c|c} \varepsilon_{\not\perp_1} & \mathbf{0} \\ \hline \mathbf{0} & \varepsilon_{\not\perp_2} \end{array}\right] \right)^\star \right) \\
& \bullet \left( \left[\begin{array}{c|c} \mathbf{T}_{\mathsf{c}_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}_2} \end{array}\right] \bullet \left[\begin{array}{c|c} \mathbf{WM}_1 & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM}_2 \end{array}\right] \right)^\star \bullet \left[\begin{array}{c} \overrightarrow{F_1} \\ \hline \overrightarrow{F_2} \end{array}\right] \\
= & \qquad \{\!\!\{ \text{ Definition of } \bullet, {}^\star \text{ and } {}^+ \text{ \& Kleene algebra } \}\!\!\} \\
& \left[\begin{array}{c|c} \overrightarrow{I_1}^t & \overrightarrow{I_2}^t \end{array}\right] \bullet \left( \left[\begin{array}{c|c} \mathbf{T}_{\perp_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{\perp_2} \end{array}\right] + \left[\begin{array}{c|c} \mathbf{WM}_1 & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM}_2 \end{array}\right] + \left[\begin{array}{c|c} \varepsilon_{\perp_1} & \mathbf{0} \\ \hline \mathbf{0} & \varepsilon_{\perp_2} \end{array}\right] \right. \\
& \qquad \left. + \left[\begin{array}{c|c} (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\perp_1} & \mathbf{0} \\ \hline \mathbf{0} & (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^+ \bullet \varepsilon_{\not\perp_2} \end{array}\right] \right)^\star \\
& \bullet \left[\begin{array}{c|c} (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star & \mathbf{0} \\ \hline \mathbf{0} & (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^\star \end{array}\right] \bullet \left[\begin{array}{c} \overrightarrow{F_1} \\ \hline \overrightarrow{F_2} \end{array}\right] \\
= & \qquad \{\!\!\{ \text{ Definition of } + \text{ and } {}^\star \text{ \& Kleene algebra } \}\!\!\}
\end{aligned}$$

$$
\begin{aligned}
&\left[\begin{array}{c|c} \overrightarrow{I_1}^t & \overrightarrow{I_2}^t \end{array}\right] \bullet \left[\begin{array}{c|c} (\mathbf{T}_{\perp_1} + \mathbf{WM}_1 + \varepsilon_{\perp_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\perp_1})^\star & \mathbf{0} \\ \hline \mathbf{0} & (\mathbf{T}_{\perp_2} + \mathbf{WM}_2 + \varepsilon_{\perp_2} + (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^+ \bullet \varepsilon_{\not\perp_2})^\star \end{array}\right] \\
&\quad \bullet \left[\begin{array}{c|c} (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star & \mathbf{0} \\ \hline \mathbf{0} & (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^\star \end{array}\right] \bullet \left[\begin{array}{c} \overrightarrow{F_1} \\ \hline \overrightarrow{F_2} \end{array}\right] \\
= \quad & \{\!\{ \text{ Definition of } \bullet \text{ \& Zero of } \cdot \text{ \& Identity of } + \text{ } \}\!\} \\
&\overrightarrow{I_1}^t \bullet (\mathbf{T}_{\perp_1} + \mathbf{WM}_1 + \varepsilon_{\perp_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\perp_1})^\star \bullet (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \\
&+ \overrightarrow{I_2}^t \bullet (\mathbf{T}_{\perp_2} + \mathbf{WM}_2 + \varepsilon_{\perp_2} + (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^+ \bullet \varepsilon_{\not\perp_2})^\star \bullet (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^\star \bullet \overrightarrow{F_2} \\
= \quad & \{\!\{ \text{ Induction hypotheses } \}\!\} \\
&q_1 + q_2
\end{aligned}
$$

For the inductive case $\cdot$, let $p := q_1 \cdot q_2$, and suppose the same inductive hypotheses as for the inductive case $+$. Define the semi-visibly pushdown automaton

$$\mathcal{A} := (S_1 \cup S_2, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, (\Gamma_1 \cup \Gamma_2) \cup \{\perp\}, \delta, I_1, F_2)$$

where $\delta := \delta_1 \cup \delta_2 \cup \{(f, \varepsilon, d; i', \perp) \mid f \in F_1 \wedge i' \in I_2 \wedge d \in (\Gamma_1 \cup \Gamma_2) \cup \{\perp\}\} \cup \mathsf{Waste}$, and

$$
\begin{aligned}
\mathsf{Waste} := \quad & \{(s, \varepsilon, d; s', \perp) \mid d \in \Gamma_2 \wedge (\exists\ d' \mid d' \in \Gamma_1 : (s, \varepsilon, d'; s', \perp) \in \delta_1)\} \\
& \cup \{(s, \varepsilon, d; s', \perp) \mid d \in \Gamma_1 \wedge (\exists\ d' \mid d' \in \Gamma_2 : (s, \varepsilon, d'; s', \perp) \in \delta_2)\} \ .
\end{aligned}
$$

It is easy to see that $\mathcal{A}$ accepts the language $\mathcal{L}(q_1 \cdot q_2)$. Note that the elements of $\mathsf{Waste}$ are added just to satisfy the property (2.3) of the definition of S-VPA.

Let $\mathcal{B} := \mathcal{B}_1, \mathcal{B}_2$ (this list of block is used since $S_1 \cap S_2 = \emptyset$ and $\Gamma_1 \cap \Gamma_2 = \emptyset$ by hypothesis, and the construction just adds $\varepsilon$-transitions). The algebraic encoding of this automaton is

$$
\left(\overrightarrow{I}, \mathbf{WM}, \left[\begin{array}{c|c} \mathbf{T}_{\mathsf{c}_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}_2} \end{array}\right], \left[\begin{array}{c|c} \mathbf{T}_{\perp_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{\perp_2} \end{array}\right], \left[\begin{array}{c|c} \varepsilon_{\perp_1} & \overrightarrow{F_1} \bullet \overrightarrow{I_2}^t \\ \hline \mathbf{0} & \varepsilon_{\perp_2} \end{array}\right], \left[\begin{array}{c|c} \varepsilon_{\not\perp_1} & \overrightarrow{F_1} \bullet \overrightarrow{I_2}^t \\ \hline \mathbf{0} & \varepsilon_{\not\perp_2} \end{array}\right], \overrightarrow{F}\right),
$$

where

$$\overrightarrow{I} := \left[\begin{array}{c} \overrightarrow{I_1} \\ \hline \overrightarrow{0} \end{array}\right] \qquad \text{and} \qquad \overrightarrow{F} := \left[\begin{array}{c} \overrightarrow{0} \\ \hline \overrightarrow{F_2} \end{array}\right] \ .$$

Once again, it is easy to see (but lengthy to prove) that

$$\vdash \mathbf{WM} = \left[\begin{array}{c|c} \mathbf{WM}_1 & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM}_2 \end{array}\right] \ . \tag{4.16}$$

The proof of (4.14) is similar to the proof of the inductive case for $+$: it is done by using (4.16), Kleene algebra, the definition of the matrix model, and the induction hypotheses.

For the inductive case $^*$, let $p := q_1^*$, and suppose the same inductive hypothesis for $q_1$ as for the inductive case $+$. Since $q_1^* = 1 + q_1 \cdot q_1^*$ by Kleene algebra and since the cases 1 and $+$ are already proved, it suffices to find a semi-visibly pushdown automaton that accepts $\mathcal{L}(q_1 \cdot q_1^*)$ and such that (4.14) is valid. Define the semi-visibly pushdown automaton

$$\mathcal{A} := (S_1, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_1 \cup \{\bot\}, \delta, I_1, F_1)$$

where

$$\delta := \delta_1 \cup \{(f, \varepsilon, d; i, \bot) \mid f \in F_1 \wedge i \in I_1 \wedge d \in \Gamma_1 \cup \{\bot\}\} \ .$$

It is easy to see that $\mathcal{A}$ accepts the language $\mathcal{L}(q_1 \cdot q_1^*)$.

Let $\mathcal{B} := \mathcal{B}_1$ (this list of block is used since the construction just adds $\varepsilon$-transitions between final states and initial states). The algebraic encoding of this automaton is

$$(\overrightarrow{I_1}, \mathbf{WM}_1, \mathbf{T}_{\mathsf{c}_1}, \mathbf{T}_{\bot_1}, \varepsilon_{\bot_1} + \overrightarrow{F_1} \bullet \overrightarrow{I_1}^t, \varepsilon_{\not\bot_1} + \overrightarrow{F_1} \bullet \overrightarrow{I_1}^t, \overrightarrow{F_1}) \ .$$

We now show (4.14).

$$\begin{aligned}
& \overrightarrow{I_1}^t \bullet (\mathbf{T}_{\bot_1} + \mathbf{WM}_1 + \varepsilon_{\bot_1} + \overrightarrow{F_1} \bullet \overrightarrow{I_1}^t + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet (\varepsilon_{\not\bot_1} + \overrightarrow{F_1} \bullet \overrightarrow{I_1}^t))^\star \bullet (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \\
= \quad & \{\!\{ \text{ Distributivity of } \cdot \text{ over } + \text{ \& Commutativity of } + \ \}\!\} \\
& \overrightarrow{I_1}^t \bullet (\overrightarrow{F_1} \bullet \overrightarrow{I_1}^t + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \overrightarrow{F_1} \bullet \overrightarrow{I_1}^t + \mathbf{T}_{\bot_1} + \mathbf{WM}_1 + \varepsilon_{\bot_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\bot_1})^\star \bullet (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \\
= \quad & \{\!\{ \text{ Kleene algebra (mainly, } 1 + q^+ = q^*) \ \}\!\} \\
& \overrightarrow{I_1}^t \bullet ((\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \bullet \overrightarrow{I_1}^t + \mathbf{T}_{\bot_1} + \mathbf{WM}_1 + \varepsilon_{\bot_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\bot_1})^\star \bullet (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \\
= \quad & \{\!\{ \text{ Kleene algebra: Denesting rule } \}\!\} \\
& \overrightarrow{I_1}^t \bullet (\mathbf{T}_{\bot_1} + \mathbf{WM}_1 + \varepsilon_{\bot_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\bot_1})^\star \\
& \bullet \left((\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \bullet \overrightarrow{I_1}^t \bullet (\mathbf{T}_{\bot_1} + \mathbf{WM}_1 + \varepsilon_{\bot_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\bot_1})^\star\right)^\star \bullet \\
& (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \\
= \quad & \{\!\{ \text{ Kleene algebra: Sliding rule } \}\!\} \\
& \overrightarrow{I_1}^t \bullet (\mathbf{T}_{\bot_1} + \mathbf{WM}_1 + \varepsilon_{\bot_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\bot_1})^\star \bullet (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \\
& \cdot \left(\overrightarrow{I_1}^t \bullet (\mathbf{T}_{\bot_1} + \mathbf{WM}_1 + \varepsilon_{\bot_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\bot_1})^\star \bullet (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1}\right)^\star \\
= \quad & \{\!\{ \text{ Induction hypothesis } \}\!\} \\
& q_1 \cdot q_1^*
\end{aligned}$$

For the inductive case of the family of operators $\mathcal{G}$, let $p := (\!|_x \mathcal{B} |\!)^y$ where $(\!|_x \mathcal{B} |\!)^y$ is an expression on a finite set of symbols $V$. Note that, for every expression $(\!|_x \mathcal{B} |\!)^y$, there exists a finite set of expressions $\{(\!|_{x'} \mathcal{C} |\!)^y\}_{x' \in I}$ where $I \subseteq V \cup \{z_{\mathbf{c}} \mid z \in V\}$ and the list $\mathcal{C}$ has the following properties:

i. the set of symbols used by $\mathcal{C}$ is $V'$ and is at most $V \cup \{z_{\mathbf{c}} \mid z \in V\}$;

ii. there is no unary block of the form $[_z\, 0\,]^{z'}$ in $\mathcal{C}^1$ where $z, z' \in V'$;

iii. there is exactly one unary block of the form $[_z\, 1\,]^{z}$ in $\mathcal{C}^1$ for each $z \in V'$;

iv. there is no other block of the form $[_z\, 1\,]^{z'}$ in $\mathcal{C}^1$ except those defined by iii;

such that

$$\vdash (\!|_x \mathcal{B} |\!)^y = (\sum x' \mid x' \in I : (\!|_{x'} \mathcal{C} |\!)^y) \ .$$

Moreover, the size of $(\sum x' \mid x' \in I : (\!|_{x'} \mathcal{C} |\!)^y)$ is polynomial in the size of $(\!|_x \mathcal{B} |\!)^y$. Intuitively, this result can be proved by doing the following:

1. Remove all unary blocks containing 0.
2. Clone the symbols from $V$ to prepare for respecting property iii. Define $\mathcal{B}'$ by the following rules:
   a. for $a \in \Sigma_{\mathbf{i}} \cup \{1\}$ and $z, z' \in V$, the unary blocks $[_z\, a\,]^{z'}$ and $[_{z_{\mathbf{c}}}\, a\,]^{z'}$ are in $(\mathcal{B}')^1$ if $[_z\, a\,]^{z'}$ is in $\mathcal{B}^1$;
   b. for $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $z, z', w, w' \in V$, the binary blocks $[_z\, c{\downarrow_{w_{\mathbf{c}}}}{\uparrow^{w'}}\, r\,]^{z'}$ and $[_{z_{\mathbf{c}}}\, c{\downarrow_{w_{\mathbf{c}}}}{\uparrow^{w'}}\, r\,]^{z'}$ are in $(\mathcal{B}')^2$ if $[_z\, c{\downarrow_{w}}{\uparrow^{w'}}\, r\,]^{z'}$ is in $\mathcal{B}^2$.

   It is easy to prove that $\vdash (\!|_x \mathcal{B} |\!)^y = (\!|_{x_{\mathbf{c}}} \mathcal{B}' |\!)^y$.
3. Add a unary block of the form $[_z\, 1\,]^{z}$ in $\mathcal{B}'$ for each $z \in V'$. The result is still equal to $(\!|_x \mathcal{B} |\!)^y$ because of step 2 that ensures that
   - the first symbol $x_{\mathbf{c}}$ is a cloned one;
   - any first symbol after a call action is a cloned one;
   - any block starting with a cloned symbol must end with a non-cloned symbol.
4. Remove all unary blocks of the form $[_z\, 1\,]^{z'}$ such that $z \neq z'$ by simulating reflexive transitive closure over unary blocks of the form $[_{z_1}\, 1\,]^{z_2}$ for every $z_1, z_2 \in V'$. Let $\mathsf{1t}^*_{\mathcal{B}'}(z)$ represents the set of symbols accessible from a symbol $z$ by going "backward" in the list $\mathcal{B}'$ through a finite number of blocks of the form $[_{z_1}\, 1\,]^{z_2}$. Define $\mathcal{C}$ by the following rules:

a. for $z \in V'$, the unary block $[_z\, 1\,]^z$ is in $\mathcal{C}^1$;

b. for $a \in \Sigma_{\mathbf{i}}$ and $z, z' \in V'$, the unary block $[_z\, a\,]^{z'}$ is in $\mathcal{C}^1$ if there exists $z'' \in V'$ such that $[_z\, a\,]^{z''}$ is in $(\mathcal{B}')^1$ and $z'' \in \mathsf{1t}^*_{\mathcal{B}'}(z')$;

c. for $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $z, z', w, w' \in V'$, the binary block $[_z\, c{\downarrow_w}{\uparrow^{w'}}\, r\,]^{z'}$ is in $\mathcal{C}^2$ if there exists $z'', w'' \in V'$ such that $[_z\, c{\downarrow_{w''}}{\uparrow^{w'}}\, r\,]^{z''}$ is in $(\mathcal{B}')^2$, $w'' \in \mathsf{1t}^*_{\mathcal{B}'}(w)$ and $z'' \in \mathsf{1t}^*_{\mathcal{B}'}(z')$.

Note that condition 4b is defined only for any $a \in \Sigma_{\mathbf{i}}$ but not for $a = 1$. At the end of this process, we have that

$$\vdash \underset{x_{\mathbf{c}}}{(\!|\, \mathcal{B}' \,|\!)}^{y} = \Big(\sum x' \mid x_{\mathbf{c}} \in \mathsf{1t}^*_{\mathcal{B}'}(x') : \underset{x'}{(\!|\, \mathcal{C} \,|\!)}^{y}\Big) \ . \tag{4.17}$$

Thus, to prove the inductive case of the family of operators $\mathcal{G}$, it suffices to prove the theorem for every expression $(\sum x' \mid x' \in I : (\!|_{x'}\, \mathcal{C}\, |\!)^y)$ defined previously. Define the semi-visibly pushdown automaton

$$\mathcal{A} := (V' \cup \{f'\}, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, V' \times \Sigma_{\mathbf{r}} \times V' \cup \{\bot\}, \delta, I, \{f'\})$$

where $f' \notin V'$ and

$$\begin{aligned}\delta := {} & \{(w, a, \lambda; w', \lambda) \mid [_w\, a\,]^{w'} \in \mathcal{C}^1\} \\ & \cup \{(w, c, \lambda; w', (z, r, z')), (z, r, (z, r, z'); z', \lambda) \mid [_w\, c{\downarrow_{w'}}{\uparrow^{z}}\, r\,]^{z'} \in \mathcal{C}^2\} \\ & \cup \{(y, \varepsilon, \bot; f', \bot)\} \ .\end{aligned}$$

It is easy to see that $\mathcal{A}$ accepts the language $\mathcal{L}((\sum x' \mid x' \in I : (\!|_{x'}\, \mathcal{C}\, |\!)^y))$. In particular, note that it accepts only well-matched words since the stack must be empty to reach the accepting state $f'$.

Let $\overrightarrow{F_1}$ be the column vector of size $|V'|$ such that 1 is in row $y$ and 0 in every other row. Let $\mathcal{C}' := \mathcal{C}, [_{f'}\, 1\,]^{f'}$. The algebraic encoding of this automaton is

$$\left(\left[\begin{array}{c} \overrightarrow{I} \\ \hline 0 \end{array}\right], \mathbf{WM}, \mathbf{T}_{\mathbf{c}}, \mathbf{0}, \left[\begin{array}{c|c} \mathbf{0} & \overrightarrow{F_1} \\ \hline 0 & 0 \end{array}\right], \mathbf{0}, \left[\begin{array}{c} \overrightarrow{0} \\ \hline 1 \end{array}\right]\right) \ .$$

It is easy to see that

$$\vdash \mathbf{WM} = \left[\begin{array}{c|c} \mathbf{WM}_1 & \overrightarrow{0} \\ \hline 0 & 1 \end{array}\right] \tag{4.18}$$

where $\mathbf{WM}_1$ is a matrix of size $|V'| \times |V'|$ containing, for each entry $j_1, j_2$, the expression $(\!|_{j_1}\, \mathcal{C}\, |\!)^{j_2}$. Also, by definition of $\mathbf{T}_{\mathbf{c}}$, it is direct that every entry containing $f'$ is 0. In other words,

$$\vdash \mathbf{T}_{\mathbf{c}} = \left[\begin{array}{c|c} \mathbf{T}_{\mathbf{c}_1} & \overrightarrow{0} \\ \hline 0 & 0 \end{array}\right] \tag{4.19}$$

where $\mathbf{T}_{\mathsf{c}_1}$ is a matrix of size $|V'| \times |V'|$ containing, for each entry $j_1, j_2$, the expression $\mathbf{T}_{\mathsf{c}}[j_1, j_2]$. We now show (4.14).

$$
\begin{aligned}
& \left[\begin{array}{c|c} \overrightarrow{I}^t & 0 \end{array}\right] \bullet \left( \mathbf{0} + \mathbf{WM} + \left[\begin{array}{c|c} \mathbf{0} & \overrightarrow{F_1} \\ \hline 0 & 0 \end{array}\right] + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \bullet \mathbf{0} \right)^\star \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^\star \bullet \left[\begin{array}{c} \overrightarrow{0} \\ \hline 1 \end{array}\right] \\
= & \quad \{\!\{ \text{Identity of } \cdot \text{ \& Zero of } + \text{ \& Equations (4.18) and (4.19)} \}\!\} \\
& \left[\begin{array}{c|c} \overrightarrow{I}^t & 0 \end{array}\right] \bullet \left( \left[\begin{array}{c|c} \mathbf{WM}_1 & \overrightarrow{0} \\ \hline 0 & 1 \end{array}\right] + \left[\begin{array}{c|c} \mathbf{0} & \overrightarrow{F_1} \\ \hline 0 & 0 \end{array}\right] \right)^\star \bullet \left( \left[\begin{array}{c|c} \mathbf{T}_{\mathsf{c}_1} & \overrightarrow{0} \\ \hline 0 & 0 \end{array}\right] \bullet \left[\begin{array}{c|c} \mathbf{WM}_1 & \overrightarrow{0} \\ \hline 0 & 1 \end{array}\right] \right)^\star \\
& \bullet \left[\begin{array}{c} \overrightarrow{0} \\ \hline 1 \end{array}\right] \\
= & \quad \{\!\{ \text{Definition of } +, \bullet \text{ and } ^\star \text{ \& Kleene algebra} \}\!\} \\
& \left[\begin{array}{c|c} \overrightarrow{I}^t & 0 \end{array}\right] \bullet \left[\begin{array}{c|c} \mathbf{WM}_1^\star & \mathbf{WM}_1^\star \bullet \overrightarrow{F_1} \\ \hline 0 & 1 \end{array}\right] \bullet \left[\begin{array}{c|c} (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star & \overrightarrow{0} \\ \hline 0 & 1 \end{array}\right] \bullet \left[\begin{array}{c} \overrightarrow{0} \\ \hline 1 \end{array}\right] \\
= & \quad \{\!\{ \text{Definition of } \bullet \text{ \& Kleene algebra} \}\!\} \\
& \overrightarrow{I}^t \bullet \mathbf{WM}_1^\star \bullet \overrightarrow{F_1} \\
= & \quad \{\!\{ \text{Lemma 4.5, property iii \& Definition of } \overrightarrow{I}, \overrightarrow{F_1} \text{ and } \mathbf{WM}_1 \text{ \& Definition of } \bullet \}\!\} \\
& (\textstyle\sum x' \mid x' \in I : (\!|_{x'} \mathcal{C} |\!)^y) \qquad \blacksquare
\end{aligned}
$$

**Theorem 4.7** (Elimination of $\varepsilon$-transitions)**.** *Let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be three disjoint finite sets such that at least one of them is nonempty. Let $\mathcal{A} := (S, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma \cup \{\bot\}, \delta, I, F)$ be a semi-visibly pushdown automaton. Let $(\overrightarrow{I}, \mathbf{WM}, \mathbf{T}_{\mathsf{c}}, \mathbf{T}_\bot, \varepsilon_\bot, \varepsilon_{\not\bot}, \overrightarrow{F})$ be the algebraic encoding of $\mathcal{A}$. Then, there exists a visibly pushdown automaton $\mathcal{A}' := (S', \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma' \cup \{\bot\}, \delta', I', F')$ having an algebraic encoding $(\overrightarrow{I'}, \mathbf{WM}', \mathbf{T}'_{\mathsf{c}}, \mathbf{T}'_\bot, \overrightarrow{F'})$ such that*

$$
\begin{aligned}
& \vdash \overrightarrow{I}^t \bullet (\mathbf{T}_\bot + \mathbf{WM} + \varepsilon_\bot + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\bot})^\star \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \\
& = \overrightarrow{I'}^t \bullet (\mathbf{T}'_\bot + \mathbf{WM}')^\star \bullet (\mathbf{T}'_{\mathsf{c}} \bullet \mathbf{WM}')^\star \bullet \overrightarrow{F'} \ .
\end{aligned} \tag{4.20}
$$

*Proof sketch (for the complete proof, see Appendix C).* This proof is done in two steps. The first step eliminates $\varepsilon$-transitions of the form $(s, \varepsilon, \bot; s', \bot)$ for all $s, s' \in S$ and the second step eliminates $\varepsilon$-transitions of the form $(s, \varepsilon, d; s', \bot)$ for all $s, s' \in S$ and $d \in \Gamma$. We give the constructions for the two steps and we present some details of the proof for the first step.

For the first step, define the function $\varepsilon\mathsf{t}^*_{\mathcal{A}} : S \to 2^S$ by

$$\varepsilon\mathsf{t}^*_{\mathcal{A}}(s) := \{s' \in S \mid \vdash \varepsilon^\star_\bot[s', s] = 1\}$$

for all $s \in S$. This function simulates the reflexive transitive closure of transitions of the form $(t_1, \varepsilon, \bot; t_2, \bot)$, but reversed: an expression $s' \in \varepsilon\mathsf{t}^*_{\mathcal{A}}(s)$ means that it is possible to start from state $s'$ and reach state $s$ by using only $\varepsilon$-transitions of the form $(t_1, \varepsilon, \bot; t_2, \bot)$. Note that every entry $(s', s)$ of $\varepsilon^\star_\bot$ can be reduced to 1 or 0 by the definition of $\varepsilon_\bot$, the definition of $^\star$, zero of $\cdot$, identity of $\cdot$ and laws: $0^* = 1$ and $1^* = 1$.

The automaton has states of the form $s_\bot$ and $s_\not\bot$ for each $s \in S$. The idea is just to encode in a state an information stating that the top of the stack is the bottom-of-stack symbol (this is represented by $s_\bot$) or not (this is represented by $s_\not\bot$). Define the semi-visibly pushdown automaton

$$\mathcal{A}'' := (\{s_\bot, s_\not\bot \mid s \in S\}, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \{d_\bot, d_\not\bot \mid d \in \Gamma\} \cup \{\bot\}, \delta', I', \{f_\bot, f_\not\bot \mid f \in F\})$$

where $I' := \{s_\bot \mid (\exists\ s' \mid s' \in I : s' \in \varepsilon\mathsf{t}^*_{\mathcal{A}}(s))\}$, and

$$\begin{aligned}
\delta' := {} & \{(s_\bot, a, \lambda; s'_\bot, \lambda) \mid a \in \Sigma_{\mathbf{i}} \wedge (\exists\ s'' \mid s'' \in \varepsilon\mathsf{t}^*_{\mathcal{A}}(s') : (s, a, \lambda; s'', \lambda) \in \delta)\} \\
& \cup\ \{(s_\not\bot, a, \lambda; s'_\not\bot, \lambda) \mid a \in \Sigma_{\mathbf{i}} \wedge (s, a, \lambda; s', \lambda) \in \delta\} \\
& \cup\ \{(s_\bot, c, \lambda; s'_\not\bot, d_\bot), (s_\not\bot, c, \lambda; s'_\not\bot, d_\not\bot) \mid c \in \Sigma_{\mathbf{c}} \wedge (s, c, \lambda; s', d) \in \delta\} \\
& \cup\ \{(s_\not\bot, r, d_\bot; s'_\bot, \lambda) \mid r \in \Sigma_{\mathbf{r}} \wedge (\exists\ s'' \mid s'' \in \varepsilon\mathsf{t}^*_{\mathcal{A}}(s') : (s, r, d; s'', \lambda) \in \delta)\} \\
& \cup\ \{(s_\not\bot, r, d_\not\bot; s'_\not\bot, \lambda) \mid r \in \Sigma_{\mathbf{r}} \wedge (s, r, d; s', \lambda) \in \delta\} \\
& \cup\ \{(s_\bot, r, \bot; s'_\bot, \bot) \mid r \in \Sigma_{\mathbf{r}} \wedge (\exists\ s'' \mid s'' \in \varepsilon\mathsf{t}^*_{\mathcal{A}}(s') : (s, r, \bot; s'', \bot) \in \delta)\} \\
& \cup\ \{(s_\not\bot, \varepsilon, d_\not\bot; s'_\bot, \bot), (s_\not\bot, \varepsilon, d_\bot; s'_\bot, \bot) \mid (\exists\ s'' \mid s'' \in \varepsilon\mathsf{t}^*_{\mathcal{A}}(s') : (s, \varepsilon, d; s'', \bot) \in \delta)\}\ .
\end{aligned}$$

The algebraic encoding of this automaton is

$$\left(\left[\begin{array}{c} \overrightarrow{I} \bullet \varepsilon^\star_\bot \\ \hline \overrightarrow{0} \end{array}\right], \mathbf{WM}', \left[\begin{array}{c|c} \mathbf{0} & \mathbf{T_c} \\ \hline \mathbf{0} & \mathbf{T_c} \end{array}\right], \left[\begin{array}{c|c} \mathbf{T}_\bot \bullet \varepsilon^\star_\bot & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} \end{array}\right], \mathbf{0}, \left[\begin{array}{c|c} \mathbf{0} & \mathbf{0} \\ \hline \varepsilon_\not\bot \bullet \varepsilon^\star_\bot & \mathbf{0} \end{array}\right], \left[\begin{array}{c} \overrightarrow{F} \\ \hline \overrightarrow{F} \end{array}\right]\right)\ .$$

First, two results are given without proof. Using these results, the proof of the first step is presented. Recall that the matrix $\mathbf{WM}$ is defined with respect to a list of blocks $\mathcal{B}$ encoding the structure of the semi-visibly pushdown automaton (except for $\varepsilon$-transitions). We first define a list of blocks $\mathcal{C}$ over labels $S \cup \{s_{\mathfrak{c}} \mid s \in S\}$ (where each label $s_{\mathfrak{c}}$ is fresh) that is very similar to $\mathcal{B}$ but does not have unary blocks $\lfloor_s\, 1\, \rceil^s$ for $s \in S$ (and binary blocks are a little different). In fact, the list of blocks $\mathcal{C}$ is defined by the following propositions:

(a) $\lfloor_{s_{\mathfrak{c}}}\, 1\, \rceil^{s_{\mathfrak{c}}} \in \mathcal{C}^1$ for all $s \in S$;

(b) $\lfloor_s\, a\, \rceil^{s'} \in \mathcal{C}^1$ and $\lfloor_{s_{\mathfrak{c}}}\, a\, \rceil^{s'_{\mathfrak{c}}} \in \mathcal{C}^1$ for all $s, s' \in S$, $a \in \Sigma_{\mathbf{i}}$ and $\lfloor_s\, a\, \rceil^{s'} \in \mathcal{B}^1$;

(c) $\lfloor_s\, c{\downarrow_{t_{\mathfrak{c}}}}{\uparrow^{t'_{\mathfrak{c}}}}\, r\, \rceil^{s'} \in \mathcal{C}^2$ and $\lfloor_{s_{\mathfrak{c}}}\, c{\downarrow_{t_{\mathfrak{c}}}}{\uparrow^{t'_{\mathfrak{c}}}}\, r\, \rceil^{s'_{\mathfrak{c}}} \in \mathcal{C}^2$ for all $s, s', t, t' \in S$, $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $\lfloor_s\, c{\downarrow_t}{\uparrow^{t'}}\, r\, \rceil^{s'} \in \mathcal{B}^2$.

Note that

$$\vdash \mathbf{WM} = \mathbf{I} + \mathbf{WM}_{\mathsf{c}} \tag{4.21}$$

where $\mathbf{WM}_{\mathsf{c}}$ is a matrix of size $|S| \times |S|$ in which each entry $(s, s') \in S \times S$ is exactly $(\!|_s \mathcal{C} |\!)^{s'}$. Also, note that

$$\vdash \mathbf{WM}' = \left[\begin{array}{c|c} (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_\perp^\star)^\star & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM} \end{array}\right] \quad . \tag{4.22}$$

We now prove that the language accepted by $\mathcal{A}''$ is the same as the one accepted by $\mathcal{A}$.

$$\begin{aligned}
&\left[\begin{array}{c|c} \overrightarrow{1}^t \bullet \varepsilon_\perp^\star & \overrightarrow{0} \end{array}\right] \bullet \left( \left[\begin{array}{c|c} \mathbf{T}_\perp \bullet \varepsilon_\perp^\star & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} \end{array}\right] + \mathbf{WM}' + \mathbf{0} \right. \\
&\qquad \left. + \left( \left[\begin{array}{c|c} \mathbf{0} & \mathbf{T}_{\mathsf{c}} \\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}} \end{array}\right] \bullet \mathbf{WM}' \right)^+ \bullet \left[\begin{array}{c|c} \mathbf{0} & \mathbf{0} \\ \hline \varepsilon_{\not\perp} \bullet \varepsilon_\perp^\star & \mathbf{0} \end{array}\right] \right)^\star \\
&\bullet \left( \left[\begin{array}{c|c} \mathbf{0} & \mathbf{T}_{\mathsf{c}} \\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}} \end{array}\right] \bullet \mathbf{WM}' \right)^\star \bullet \left[\begin{array}{c} \overrightarrow{F} \\ \hline \overrightarrow{F} \end{array}\right] \\
= &\qquad \{\!\!\{ \text{ Identity of } + \text{ \& Equation (4.22) } \}\!\!\} \\
&\left[\begin{array}{c|c} \overrightarrow{1}^t \bullet \varepsilon_\perp^\star & \overrightarrow{0} \end{array}\right] \bullet \left( \left[\begin{array}{c|c} \mathbf{T}_\perp \bullet \varepsilon_\perp^\star & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} \end{array}\right] + \left[\begin{array}{c|c} (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_\perp^\star)^\star & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM} \end{array}\right] \right. \\
&\qquad \left. + \left( \left[\begin{array}{c|c} \mathbf{0} & \mathbf{T}_{\mathsf{c}} \\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}} \end{array}\right] \bullet \left[\begin{array}{c|c} (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_\perp^\star)^\star & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM} \end{array}\right] \right)^+ \bullet \left[\begin{array}{c|c} \mathbf{0} & \mathbf{0} \\ \hline \varepsilon_{\not\perp} \bullet \varepsilon_\perp^\star & \mathbf{0} \end{array}\right] \right)^\star \\
&\bullet \left( \left[\begin{array}{c|c} \mathbf{0} & \mathbf{T}_{\mathsf{c}} \\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}} \end{array}\right] \bullet \left[\begin{array}{c|c} (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_\perp^\star)^\star & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM} \end{array}\right] \right)^\star \bullet \left[\begin{array}{c} \overrightarrow{F} \\ \hline \overrightarrow{F} \end{array}\right] \\
= &\qquad \{\!\!\{ \text{ Definition of } \bullet, {}^+ \text{ and } {}^\star \text{ \& Kleene algebra } \}\!\!\} \\
&\left[\begin{array}{c|c} \overrightarrow{1}^t \bullet \varepsilon_\perp^\star & \overrightarrow{0} \end{array}\right] \bullet \left( \left[\begin{array}{c|c} \mathbf{T}_\perp \bullet \varepsilon_\perp^\star & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} \end{array}\right] + \left[\begin{array}{c|c} (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_\perp^\star)^\star & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM} \end{array}\right] \right. \\
&\qquad \left. + \left[\begin{array}{c|c} \mathbf{0} & (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \\ \hline \mathbf{0} & (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \end{array}\right] \bullet \left[\begin{array}{c|c} \mathbf{0} & \mathbf{0} \\ \hline \varepsilon_{\not\perp} \bullet \varepsilon_\perp^\star & \mathbf{0} \end{array}\right] \right)^\star \\
&\bullet \left[\begin{array}{c|c} \mathbf{I} & (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \\ \hline \mathbf{0} & (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^\star \end{array}\right] \bullet \left[\begin{array}{c} \overrightarrow{F} \\ \hline \overrightarrow{F} \end{array}\right] \\
= &\qquad \{\!\!\{ \text{ Definition of } \bullet \text{ and } + \text{ } \}\!\!\} \\
&\left[\begin{array}{c|c} \overrightarrow{1}^t \bullet \varepsilon_\perp^\star & \overrightarrow{0} \end{array}\right] \bullet \left[\begin{array}{c|c} \mathbf{T}_\perp \bullet \varepsilon_\perp^\star + (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_\perp^\star)^\star + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp} \bullet \varepsilon_\perp^\star & \mathbf{0} \\ \hline (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp} \bullet \varepsilon_\perp^\star & \mathbf{WM} \end{array}\right]^\star \\
&\bullet \left[\begin{array}{c|c} \mathbf{I} & (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \\ \hline \mathbf{0} & (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^\star \end{array}\right] \bullet \left[\begin{array}{c} \overrightarrow{F} \\ \hline \overrightarrow{F} \end{array}\right]
\end{aligned}$$

$$
\begin{array}{ll}
= & \{\!\{ \text{ Definition of } ^\star \text{ \& Kleene algebra } \}\!\} \\
& \left[\begin{array}{c|c} \overrightarrow{I}^t \bullet \varepsilon_\perp^\star & \overrightarrow{0} \end{array}\right] \bullet \left[\begin{array}{c|c} \begin{array}{l} (\mathbf{T}_\perp \bullet \varepsilon_\perp^\star + (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_\perp^\star)^\star \\ \quad + \ (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp} \bullet \varepsilon_\perp^\star)^\star \end{array} & \mathbf{0} \\ \hline \begin{array}{l} \mathbf{WM}^\star \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp} \bullet \varepsilon_\perp^\star \\ \bullet \ (\mathbf{T}_\perp \bullet \varepsilon_\perp^\star + (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_\perp^\star)^\star \\ \quad + \ (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp} \bullet \varepsilon_\perp^\star)^\star \end{array} & \mathbf{WM}^\star \end{array}\right] \\
& \bullet \left[\begin{array}{c|c} \mathbf{I} & (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \\ \hline \mathbf{0} & (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^\star \end{array}\right] \bullet \left[\begin{array}{c} \overrightarrow{F} \\ \hline \overrightarrow{F} \end{array}\right] \\
= & \{\!\{ \text{ Definition of } \bullet \text{ \& Kleene algebra } \}\!\} \\
& \left[\begin{array}{c|c} \overrightarrow{I}^t \bullet \varepsilon_\perp^\star \bullet (\mathbf{T}_\perp \bullet \varepsilon_\perp^\star + (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_\perp^\star)^\star + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp} \bullet \varepsilon_\perp^\star)^\star & \overrightarrow{0} \end{array}\right] \\
& \bullet \left[\begin{array}{c} (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \\ \hline (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \end{array}\right] \\
= & \{\!\{ \text{ Definition of } \bullet \text{ \& Kleene algebra } \}\!\} \\
& \overrightarrow{I}^t \bullet \varepsilon_\perp^\star \bullet (\mathbf{T}_\perp \bullet \varepsilon_\perp^\star + (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_\perp^\star)^\star + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp} \bullet \varepsilon_\perp^\star)^\star \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \\
= & \{\!\{ \text{ Kleene algebra: } (p^* + q)^* = (p + q)^* \ \}\!\} \\
& \overrightarrow{I}^t \bullet \varepsilon_\perp^\star \bullet (\mathbf{T}_\perp \bullet \varepsilon_\perp^\star + \mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_\perp^\star + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp} \bullet \varepsilon_\perp^\star)^\star \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \\
= & \{\!\{ \text{ Distributivity of } \cdot \text{ over } + \ \}\!\} \\
& \overrightarrow{I}^t \bullet \varepsilon_\perp^\star \bullet ((\mathbf{T}_\perp + \mathbf{WM}_{\mathsf{c}} + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp}) \bullet \varepsilon_\perp^\star)^\star \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \\
= & \{\!\{ \text{ Denesting rule } \}\!\} \\
& \overrightarrow{I}^t \bullet (\mathbf{T}_\perp + \mathbf{WM}_{\mathsf{c}} + \varepsilon_\perp + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \\
= & \{\!\{ \text{ Kleene algebra: } p^* = (1 + p)^* \ \}\!\} \\
& \overrightarrow{I}^t \bullet (\mathbf{T}_\perp + \mathbf{I} + \mathbf{WM}_{\mathsf{c}} + \varepsilon_\perp + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \\
= & \{\!\{ \text{ Equation (4.21) } \}\!\} \\
& \overrightarrow{I}^t \bullet (\mathbf{T}_\perp + \mathbf{WM} + \varepsilon_\perp + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F}
\end{array}
$$

For the second step, suppose that we have a S-VPA $\mathcal{A} := (S, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma \cup \{\perp\}, \delta, I, F)$ such that $\delta$ does not contain $\varepsilon$-transitions of the form $(s, \varepsilon, \perp; s', \perp)$ for all $s, s' \in S$. This is always possible because of the first step of the proof. Let $(\overrightarrow{I}, \mathbf{WM}, \mathbf{T}_{\mathsf{c}}, \mathbf{T}_\perp, \varepsilon_\perp, \varepsilon_{\not\perp}, \overrightarrow{F})$ be the algebraic encoding of $\mathcal{A}$. We construct a S-VPA that does not have $\varepsilon$-transitions of the form $(s, \varepsilon, d; s', \perp)$ for all $s, s' \in S$ and $d \in \Gamma$.

The automaton has states of the form $s_{\mathfrak{a}_\perp}$, $s_{\mathfrak{a}_{\not\perp}}$ and $s_{\mathfrak{p}_{\not\perp}}$ for each $s \in S$. The idea is just to encode in a state an information stating if the stack is "allowed" to be read normally and is empty (this is represented by $s_{\mathfrak{a}_\perp}$) or if the stack is "allowed" to be read normally and is nonempty (this is represented by $s_{\mathfrak{a}_{\not\perp}}$) or if it must be "protected" when reading symbols from the nonempty stack (this is represented by $s_{\mathfrak{p}_{\not\perp}}$). A state $s_{\mathfrak{p}_{\not\perp}}$ protecting the reading of the stack is just a way of saying that the values of the symbols

on the stack are irrelevant. A value of a symbol on the stack becomes irrelevant when simulating a transition of the form $(t, \varepsilon, d; t', \bot)$ which protects the stack from being read. The idea is that any transition of the form $(t, r, \bot; t', \bot)$ is simulated in $t_{\mathfrak{p}_\not}$ by saturation of the transitions over the possible stack symbols.

Define the visibly pushdown automaton

$$\begin{aligned}\mathcal{A}' := (&\{s_{\mathfrak{a}_\bot}, s_{\mathfrak{a}_\not}, s_{\mathfrak{p}_\not} \mid s \in S\}, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \{\mathfrak{a}_\bot, \mathfrak{a}_\not, \mathfrak{p}_\not\} \times \Gamma \cup \{\bot\}, \delta', \{i_{\mathfrak{a}_\bot} \mid i \in I\},\\ &\{f_{\mathfrak{a}_\bot}, f_{\mathfrak{a}_\not}, f_{\mathfrak{p}_\not} \mid f \in F\})\end{aligned}$$

where

$$\begin{aligned}\delta' = \{&(s_{\mathfrak{a}_\bot}, a, \lambda; s'_{\mathfrak{a}_\bot}, \lambda), (s_{\mathfrak{a}_\not}, a, \lambda; s'_{\mathfrak{a}_\not}, \lambda), (s_{\mathfrak{p}_\not}, a, \lambda; s'_{\mathfrak{p}_\not}, \lambda) \mid a \in \Sigma_{\mathbf{i}}\\ &\quad \wedge\ (s, a, \lambda; s', \lambda) \in \delta\}\\ \cup\ &\{(s_{\mathfrak{a}_\not}, a, \lambda; s'_{\mathfrak{p}_\not}, \lambda) \mid a \in \Sigma_{\mathbf{i}}\\ &\quad \wedge\ (\exists\ s'' \mid (\forall\ d \mid d \in \Gamma : (s'', \varepsilon, d; s', \bot) \in \delta) : (s, a, \lambda; s'', \lambda) \in \delta)\}\\ \cup\ &\{(s_{\mathfrak{a}_\bot}, c, \lambda; s'_{\mathfrak{a}_\not}, (\mathfrak{a}_\bot, d)), (s_{\mathfrak{a}_\not}, c, \lambda; s'_{\mathfrak{a}_\not}, (\mathfrak{a}_\not, d)), (s_{\mathfrak{p}_\not}, c, \lambda; s'_{\mathfrak{a}_\not}, (\mathfrak{p}_\not, d)) \mid c \in \Sigma_{\mathbf{c}}\\ &\quad \wedge\ (s, c, \lambda; s', d) \in \delta\}\\ \cup\ &\{(s_{\mathfrak{a}_\bot}, c, \lambda; s'_{\mathfrak{p}_\not}, (\mathfrak{a}_\bot, d)), (s_{\mathfrak{a}_\not}, c, \lambda; s'_{\mathfrak{p}_\not}, (\mathfrak{a}_\not, d)), (s_{\mathfrak{p}_\not}, c, \lambda; s'_{\mathfrak{p}_\not}, (\mathfrak{p}_\not, d)) \mid c \in \Sigma_{\mathbf{c}}\\ &\quad \wedge\ (\exists\ s'' \mid (s'', \varepsilon, d; s', \bot) \in \delta : (s, c, \lambda; s'', d) \in \delta)\}\\ \cup\ &\{(s_{\mathfrak{a}_\not}, r, (\mathfrak{a}_\bot, d); s'_{\mathfrak{a}_\bot}, \lambda), (s_{\mathfrak{a}_\not}, r, (\mathfrak{a}_\not, d); s'_{\mathfrak{a}_\not}, \lambda), (s_{\mathfrak{a}_\not}, r, (\mathfrak{p}_\not, d); s'_{\mathfrak{p}_\not}, \lambda) \mid r \in \Sigma_{\mathbf{r}}\\ &\quad \wedge\ (s, r, d; s', \lambda) \in \delta\}\\ \cup\ &\{(s_{\mathfrak{a}_\not}, r, (\mathfrak{a}_\not, d); s'_{\mathfrak{p}_\not}, \lambda) \mid r \in \Sigma_{\mathbf{r}}\\ &\quad \wedge\ (\exists\ s'' \mid (\forall\ d' \mid d' \in \Gamma : (s'', \varepsilon, d'; s', \bot) \in \delta) : (s, r, d; s'', \lambda) \in \delta)\}\\ \cup\ &\{(s_{\mathfrak{a}_\bot}, r, \bot; s'_{\mathfrak{a}_\bot}, \bot) \mid r \in \Sigma_{\mathbf{r}} \wedge (s, r, \bot; s', \bot) \in \delta\}\\ \cup\ &\{(s_{\mathfrak{p}_\not}, r, (\mathfrak{a}_\bot, d); s'_{\mathfrak{a}_\bot}, \lambda) \mid r \in \Sigma_{\mathbf{r}} \wedge d \in \Gamma \wedge (s, r, \bot; s', \bot) \in \delta\}\\ \cup\ &\{(s_{\mathfrak{p}_\not}, r, (\mathfrak{a}_\not, d); s'_{\mathfrak{p}_\not}, \lambda), (s_{\mathfrak{p}_\not}, r, (\mathfrak{p}_\not, d); s'_{\mathfrak{p}_\not}, \lambda) \mid r \in \Sigma_{\mathbf{r}} \wedge d \in \Gamma\\ &\quad \wedge\ (s, r, \bot; s', \bot) \in \delta\}\ .\end{aligned}$$

The automaton $\mathcal{A}'$ accepts the same language as $\mathcal{A}$. In other words, equation (4.20) is valid. ■

**Theorem 4.8** (Determinization of visibly pushdown automata)**.** *Let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be three disjoint finite sets such that at least one of them is nonempty. Let*

$$\mathcal{A} := (S, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma \cup \{\bot\}, \delta, I, F)$$

*be a visibly pushdown automaton. Let $(\overrightarrow{I}, \mathbf{WM}, \mathbf{T_c}, \mathbf{T}_\bot, \overrightarrow{F})$ be the algebraic encoding of $\mathcal{A}$. Then, there exists a* deterministic *visibly pushdown automaton*

$$\mathcal{A}' := (S', \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma' \cup \{\bot\}, \delta', I', F')$$

*having an algebraic encoding $(\overrightarrow{I'}, \mathbf{WM}', \mathbf{T}'_{\mathbf{c}}, \mathbf{T}'_\bot, \overrightarrow{F'})$ such that*

$$\vdash \overrightarrow{I}^t \bullet (\mathbf{T}_\bot + \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \quad = \quad \overrightarrow{I'}^t \bullet (\mathbf{T}'_\bot + \mathbf{WM}')^\star \bullet (\mathbf{T}'_{\mathbf{c}} \bullet \mathbf{WM}')^\star \bullet \overrightarrow{F'}\ . \quad (4.23)$$

*Proof sketch (for the complete proof, see Appendix D).* The construction used here is a simplification of Alur and Madhusudan's construction [1]: the deterministic automaton $\mathcal{A}'$ just uses states of the form $T$ where $T \subseteq S \times S$ is a binary relation between states of $\mathcal{A}$. Intuitively, the range of the relation $T$ represents the current set of reachable states of $S$. This is the "standard" component of the usual subset construction of finite automata. On the other hand, the domain of $T$ represents the set of states reached *just after reading the last possibly pending call in the current run* (except for the initial state in which an identity function of $I$ is used). The component $T$ is used to postpone the evaluation of push actions on the stack until their associated pop action occurs (if there is such a pop action). Note that only the *evaluation* of the push action on the stack is postponed and not the *reading* of the call action. Note also that if there is no pop action associated to a push action, then the evaluation of this push action on the stack is not important since it is a pending call (it will never be used).

To understand the component $T$ more clearly, take a word $w = a_1 a_2 \dots a_i \dots a_j \dots a_k$ where $a_i$ is a call action and $a_j$ is its associated return action. Since $\mathcal{A}'$ is a deterministic pushdown automaton, there is one and only one run of $\mathcal{A}'$ on $w$. Suppose that we stop the run just after the action $a_j$. Name the state of $\mathcal{A}'$ just after the action $a_j$ by $T_{j+1}$. Each pair $(s, s') \in T_{j+1}$ represents that it is possible to start $\mathcal{A}$ in the state $s$ with an empty stack ($\bot$), run $\mathcal{A}$ on the affix $a_i \dots a_j$ and end the run in the state $s'$ of $\mathcal{A}$ with an empty stack ($\bot$).

We now express this construction formally. Let

$$\begin{array}{rcll} S' & := & 2^{S \times S}, & \\ \mathsf{ran}(T) & := & \{s' \in S \mid (\exists\ s \mid : (s, s') \in T)\} & \text{for } T \subseteq S \times S, \\ \mathsf{Id}_R & := & \{(s, s) \in S \times S \mid s \in R\} & \text{for } R \subseteq S. \end{array}$$

Construct the following deterministic visibly pushdown automaton:

$$\mathcal{A}' := (S', \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, S' \times \Sigma_{\mathbf{c}} \cup \{\bot\}, \delta', \{\mathsf{Id}_I\}, \{T \in S' \mid \mathsf{ran}(T) \cap F \neq \emptyset\})$$

where $\delta'$ is defined by the set of all the following transitions:

- a transition $(T, a, \lambda; T', \lambda)$ for each $a \in \Sigma_{\mathbf{i}}$ and $T \in S'$ where

$$T' := \{(s, s') \in S \times S \mid (\exists\ s'' \mid (s, s'') \in T : (s'', a, \lambda; s', \lambda) \in \delta)\}\ ;$$

- a transition $(T, c, \lambda; \mathsf{Id}_{R'}, (T, c))$ for each $c \in \Sigma_{\mathbf{c}}$ and $T \in S'$ where

$$R' := \{s' \in S \mid (\exists\ s, d \mid s \in \mathsf{ran}(T) \wedge d \in \Gamma : (s, c, \lambda; s', d) \in \delta)\}\ ;$$

- a transition $(T, r, \bot; T', \bot)$ for each $r \in \Sigma_{\mathbf{r}}$ and $T \in S'$ where

$$T' := \{(s, s') \in S \times S \mid (\exists\ s'' \mid (s, s'') \in T : (s'', r, \bot; s', \bot) \in \delta)\}\ ;$$

- a transition $(T, r, (T'', c); T', \lambda)$ for each $r \in \Sigma_{\mathbf{r}}$, $T \in S'$ and $(T'', c) \in S' \times \Sigma_{\mathbf{c}}$ where

$$T' := \{(s, s') \in S \times S \mid (\exists\ s'' \mid (s, s'') \in T'' : (s'', s') \in \mathsf{Update})\}$$

and

$$\mathsf{Update} := \{(s, s') \in S \times S \mid (\exists s_1, s_2, d \mid (s_1, s_2) \in T \wedge d \in \Gamma : (s, c, \lambda; s_1, d) \in \delta \wedge (s_2, r, d; s', \lambda) \in \delta)\}\ .$$

We now show (4.23). To do this, we use a projection between states of $2^{S \times S}$ and states of $S$. The projection is in fact the composition of two simple projections $\mathbf{X}$ and $\mathbf{Y}$ expressed as matrices. Let $\mathbf{X}$ be a matrix of size $|2^{S \times S}| \times |S \times S|$ defined for each $T \subseteq S \times S$ and $s, s_2 \in S$ by

$$\mathbf{X}[T, (s_2, s)] := \begin{cases} 1 & \text{if } (s_2, s) \in T, \\ 0 & \text{otherwise.} \end{cases}$$

Let also $\mathbf{Y}$ be a matrix of size $|S \times S| \times |S|$ defined for each $s, s', s_2 \in S$ by

$$\mathbf{Y}[(s_2, s), s'] := \begin{cases} 1 & \text{if } s = s', \\ 0 & \text{otherwise.} \end{cases}$$

The projection $\mathbf{X} \bullet \mathbf{Y}$ is split in two because it helps a lot, for well-matched words, to have an "intermediate" level in which the component $T$ is "explicit".

The projection $\mathbf{X} \bullet \mathbf{Y}$ is useful since it suffices to prove that

$$\vdash \overrightarrow{I'}^t \bullet \mathbf{X} \bullet \mathbf{Y} = \overrightarrow{I}^t\ , \tag{4.24}$$

$$\vdash \overrightarrow{F'} = \mathbf{X} \bullet \mathbf{Y} \bullet \overrightarrow{F}\ , \tag{4.25}$$

$$\vdash \mathbf{T}'_{\mathsf{c}} \bullet \mathbf{X} \bullet \mathbf{Y} = \mathbf{X} \bullet \mathbf{Y} \bullet \mathbf{T}_{\mathsf{c}}\ , \tag{4.26}$$

$$\vdash \mathbf{T}'_{\bot} \bullet \mathbf{X} \bullet \mathbf{Y} = \mathbf{X} \bullet \mathbf{Y} \bullet \mathbf{T}_{\bot}\ , \tag{4.27}$$

$$\vdash \mathbf{WM}' \bullet \mathbf{X} \bullet \mathbf{Y} = \mathbf{X} \bullet \mathbf{Y} \bullet \mathbf{WM}\ , \tag{4.28}$$

and (4.23) follows easily,

$$\begin{aligned} & \overrightarrow{I'}^t \bullet (\mathbf{T}'_{\bot} + \mathbf{WM}')^\star \bullet (\mathbf{T}'_{\mathsf{c}} \bullet \mathbf{WM}')^\star \bullet \overrightarrow{F'} \\ = & \qquad \{\!\!\{ \text{ Equation (4.25) } \}\!\!\} \end{aligned}$$

$$
\begin{aligned}
& \overrightarrow{I'}^{t} \bullet (\mathbf{T}'_{\perp} + \mathbf{WM}')^{\star} \bullet (\mathbf{T}'_{\mathsf{c}} \bullet \mathbf{WM}')^{\star} \bullet \mathbf{X} \bullet \mathbf{Y} \bullet \overrightarrow{F} \\
= & \quad \{\!\{ \text{ By (4.28) and (4.26), it is direct that } \mathbf{T}'_{\mathsf{c}} \bullet \mathbf{WM}' \bullet \mathbf{X} \bullet \mathbf{Y} = \mathbf{X} \bullet \mathbf{Y} \bullet \mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM}. \\
& \quad \text{\& Kleene algebra: Bisimulation rule } \}\!\} \\
& \overrightarrow{I'}^{t} \bullet (\mathbf{T}'_{\perp} + \mathbf{WM}')^{\star} \bullet \mathbf{X} \bullet \mathbf{Y} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet \overrightarrow{F} \\
= & \quad \{\!\{ \text{ By (4.28), (4.27) and Kleene algebra, it is direct that } (\mathbf{T}'_{\perp} + \mathbf{WM}') \bullet \mathbf{X} \bullet \\
& \quad \mathbf{Y} = \mathbf{X} \bullet \mathbf{Y} \bullet (\mathbf{T}_{\perp} + \mathbf{WM}). \text{ \& Kleene algebra: Bisimulation rule } \}\!\} \\
& \overrightarrow{I'}^{t} \bullet \mathbf{X} \bullet \mathbf{Y} \bullet (\mathbf{T}_{\perp} + \mathbf{WM})^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet \overrightarrow{F} \\
= & \quad \{\!\{ \text{ Equation (4.24) } \}\!\} \\
& \overrightarrow{I}^{t} \bullet (\mathbf{T}_{\perp} + \mathbf{WM})^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet \overrightarrow{F} \ .
\end{aligned}
$$

The proof of (4.24) to (4.28) are straightforward. Note that the proof of (4.28) essentially uses axioms (3.16) and (3.17) since we are proving a bisimulation between the two automata. This is our first use of these axioms since the beginning of this dissertation. ■

**Theorem 4.9** (Synchronization of two deterministic VPAs)**.** *Let* $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ *and* $\Sigma_{\mathbf{r}}$ *be three disjoint finite sets such that at least one of them is nonempty. Let* $\mathcal{A}_1 := (S_1, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_1 \cup \{\perp\}, \delta_1, I_1, F_1)$ *be a deterministic visibly pushdown automaton and*

$$(\overrightarrow{I_1}, \mathbf{WM}_1, \mathbf{T}_{\mathsf{c}_1}, \mathbf{T}_{\perp_1}, \overrightarrow{F_1})$$

*its algebraic encoding. Let* $\mathcal{A}_2 := (S_2, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_2 \cup \{\perp\}, \delta_2, I_2, F_2)$ *be another deterministic visibly pushdown automaton and*

$$(\overrightarrow{I_2}, \mathbf{WM}_2, \mathbf{T}_{\mathsf{c}_2}, \mathbf{T}_{\perp_2}, \overrightarrow{F_2})$$

*its algebraic encoding. Then, there exist two deterministic visibly pushdown automata*

$$\mathcal{A}'_1 := (S, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma \cup \{\perp\}, \delta, I, F'_1) \qquad \textit{and} \qquad \mathcal{A}'_2 := (S, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma \cup \{\perp\}, \delta, I, F'_2)$$

*having respectively algebraic encodings*

$$(\overrightarrow{I}, \mathbf{WM}, \mathbf{T}_{\mathsf{c}}, \mathbf{T}_{\perp}, \overrightarrow{F'_1}) \qquad \textit{and} \qquad (\overrightarrow{I}, \mathbf{WM}, \mathbf{T}_{\mathsf{c}}, \mathbf{T}_{\perp}, \overrightarrow{F'_2})$$

*such that* $\mathcal{A}'_1$ *and* $\mathcal{A}'_2$ *differ only in their accepting states* ($F'_1$ *and* $F'_2$ *are not necessarily equal) and are such that*

$$
\begin{aligned}
& \vdash \overrightarrow{I_1}^{t} \bullet (\mathbf{T}_{\perp_1} + \mathbf{WM}_1)^{\star} \bullet (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^{\star} \bullet \overrightarrow{F_1} \\
& = \ \overrightarrow{I}^{t} \bullet (\mathbf{T}_{\perp} + \mathbf{WM})^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet \overrightarrow{F'_1} \ ,
\end{aligned}
\tag{4.29}
$$

$$
\begin{aligned}
& \vdash \overrightarrow{I_2}^{t} \bullet (\mathbf{T}_{\perp_2} + \mathbf{WM}_2)^{\star} \bullet (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^{\star} \bullet \overrightarrow{F_2} \\
& = \ \overrightarrow{I}^{t} \bullet (\mathbf{T}_{\perp} + \mathbf{WM})^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet \overrightarrow{F'_2} \ .
\end{aligned}
\tag{4.30}
$$

*Proof sketch (for the complete proof, see Appendix E).* Without loss of generality, suppose that $\Gamma_1 \neq \emptyset$ and $\Gamma_2 \neq \emptyset$. The idea of the construction is to use the synchronous product of visibly pushdown automata.

Define the first deterministic visibly pushdown automaton by

$$(S_1 \times S_2, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_1 \times \Gamma_2 \cup \{\bot\}, \delta, I_1 \times I_2, F_1 \times S_2)$$

and the second one by

$$(S_1 \times S_2, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_1 \times \Gamma_2 \cup \{\bot\}, \delta, I_1 \times I_2, S_1 \times F_2) \ ,$$

where $\delta$ is defined as the set of all the following transitions:

- a transition $((s,s'),a,\lambda;(t,t'),\lambda)$ for all pairs $(s,s') \in S \times S'$, internal actions $a \in \Sigma_{\mathbf{i}}$ and transitions $(s,a,\lambda;t,\lambda) \in \delta_1$ and $(s',a,\lambda;t',\lambda) \in \delta_2$;
- a transition $((s,s'),c,\lambda;(t,t'),(d,d'))$ for all pairs $(s,s') \in S \times S'$, call actions $c \in \Sigma_{\mathbf{c}}$ and transitions $(s,c,\lambda;t,d) \in \delta_1$ and $(s',c,\lambda;t',d') \in \delta_2$;
- a transition $((s,s'),r,(d,d');(t,t'),\lambda)$ for all pairs $(s,s') \in S \times S'$, return actions $r \in \Sigma_{\mathbf{r}}$, stack symbols $d \in \Gamma_1$, $d' \in \Gamma_2$ and transitions $(s,r,d;t,\lambda) \in \delta_1$ and $(s',r,d';t',\lambda) \in \delta_2$;
- a transition $((s,s'),r,\bot;(t,t'),\bot)$ for all pairs $(s,s') \in S \times S'$, return actions $r \in \Sigma_{\mathbf{r}}$ and transitions $(s,r,\bot;t,\bot) \in \delta_1$ and $(s',r,\bot;t',\bot) \in \delta_2$.

We only show (4.29), because the proof of (4.30) is similar.

To do this, we use a projection between states of $S_1 \times S_2$ and states of $S_1$. The projection is expressed as a matrix. Let $\mathbf{X}$ be a matrix of size $|S_1 \times S_2| \times |S_1|$ defined for each $s,t \in S_1$ and $s' \in S_2$ by

$$\mathbf{X}[(s,s'),t] := \begin{cases} 1 & \text{if } s = t, \\ 0 & \text{otherwise.} \end{cases}$$

The projection $\mathbf{X}$ is useful since it suffices to prove that

$$\begin{aligned}
&\vdash \overrightarrow{I}^t \bullet \mathbf{X} = \overrightarrow{I_1}^t \ , \\
&\vdash \overrightarrow{F_1'} = \mathbf{X} \bullet \overrightarrow{F_1} \ , \\
&\vdash \mathbf{T}_{\mathsf{c}} \bullet \mathbf{X} = \mathbf{X} \bullet \mathbf{T}_{\mathsf{c}_1} \ , \\
&\vdash \mathbf{T}_{\bot} \bullet \mathbf{X} = \mathbf{X} \bullet \mathbf{T}_{\bot_1} \ , \\
&\vdash \mathbf{WM} \bullet \mathbf{X} = \mathbf{X} \bullet \mathbf{WM}_1 \ .
\end{aligned}$$

and (4.29) follows easily. In fact, the rest of the proof is similar in spirit to the proof of Theorem 4.8. ■

We are now ready to prove the completeness of the equational theory of VPKA over the language model under interpretation $\mathcal{L}$.

*Proof of Theorem 4.2.* We suppose $\mathcal{L}(p) = \mathcal{L}(q)$ and we prove $\vdash p = q$. By the generalization of Kleene's representation theorem for semi-visibly pushdown automata (Theorem 4.6), there exists semi-visibly pushdown automata

$$\mathcal{A}_1 := (S_1, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_1 \cup \{\bot\}, \delta_1, I_1, F_1)$$

and

$$\mathcal{A}_2 := (S_2, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_2 \cup \{\bot\}, \delta_2, I_2, F_2)$$

that accept respectively the languages $\mathcal{L}(p)$ and $\mathcal{L}(q)$ and such that the algebraic encoding of $\mathcal{A}_1$ and $\mathcal{A}_2$ by the structures

$$(\overrightarrow{I_1}, \mathbf{WM}_1, \mathbf{T}_{\mathsf{c}_1}, \mathbf{T}_{\bot_1}, \varepsilon_{\bot_1}, \varepsilon_{\not\bot_1}, \overrightarrow{F_1}) \qquad \text{and} \qquad (\overrightarrow{I_2}, \mathbf{WM}_2, \mathbf{T}_{\mathsf{c}_2}, \mathbf{T}_{\bot_2}, \varepsilon_{\bot_2}, \varepsilon_{\not\bot_2}, \overrightarrow{F_2})$$

satisfies

$$\begin{aligned} \vdash p &= \overrightarrow{I_1}^t \bullet (\mathbf{T}_{\bot_1} + \mathbf{WM}_1 + \varepsilon_{\bot_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\bot_1})^\star \bullet (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \ , \\ \vdash q &= \overrightarrow{I_2}^t \bullet (\mathbf{T}_{\bot_2} + \mathbf{WM}_2 + \varepsilon_{\bot_2} + (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^+ \bullet \varepsilon_{\not\bot_2})^\star \bullet (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^\star \bullet \overrightarrow{F_2} \ . \end{aligned}$$

Then, by the elimination of $\varepsilon$-transitions (Theorem 4.7), by the determinization of visibly pushdown automata (Theorem 4.8) and by the synchronization of two deterministic visibly pushdown automata (Theorem 4.9), there exists two deterministic visibly pushdown automata

$$\mathcal{A}'_1 := (S, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma \cup \{\bot\}, \delta, I, F'_1) \qquad \text{and} \qquad \mathcal{A}'_2 := (S, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma \cup \{\bot\}, \delta, I, F'_2)$$

having respectively algebraic encodings

$$(\overrightarrow{I}, \mathbf{WM}, \mathbf{T}_{\mathsf{c}}, \mathbf{T}_{\bot}, \overrightarrow{F'_1}) \qquad \text{and} \qquad (\overrightarrow{I}, \mathbf{WM}, \mathbf{T}_{\mathsf{c}}, \mathbf{T}_{\bot}, \overrightarrow{F'_2})$$

such that these automata differ only in their accepting states and that

$$\begin{aligned} &\vdash \overrightarrow{I_1}^t \bullet (\mathbf{T}_{\bot_1} + \mathbf{WM}_1 + \varepsilon_{\bot_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\bot_1})^\star \bullet (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \\ &\quad = \overrightarrow{I}^t \bullet (\mathbf{T}_{\bot} + \mathbf{WM})^\star \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F'_1} \ , \\ &\vdash \overrightarrow{I_2}^t \bullet (\mathbf{T}_{\bot_2} + \mathbf{WM}_2 + \varepsilon_{\bot_2} + (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^+ \bullet \varepsilon_{\not\bot_2})^\star \bullet (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^\star \bullet \overrightarrow{F_2} \\ &\quad = \overrightarrow{I}^t \bullet (\mathbf{T}_{\bot} + \mathbf{WM})^\star \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F'_2} \ . \end{aligned}$$

Now, let us get rid of *inaccessible states* in $F_1'$ and $F_2'$. Let

$$\mathsf{IS} := \{s \in S \mid \vdash \big(\overrightarrow{I}^t \bullet (\mathbf{T}_\perp + \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star\big)[1, s] = 0\} \ .$$

It is easy to see that $\mathsf{IS}$ is the set of inaccessible states in both $\mathcal{A}_1'$ and $\mathcal{A}_2'$. Also, by Kleene algebra,

$$\begin{aligned} \vdash \overrightarrow{I}^t \bullet (\mathbf{T}_\perp + \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F_1'} \\ = \overrightarrow{I}^t \bullet (\mathbf{T}_\perp + \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F_1' \backslash \mathsf{IS}} \ , \\ \vdash \overrightarrow{I}^t \bullet (\mathbf{T}_\perp + \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F_2'} \\ = \overrightarrow{I}^t \bullet (\mathbf{T}_\perp + \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F_2' \backslash \mathsf{IS}} \ . \end{aligned}$$

Let us summarize what has been done so far. The proof started with two VPREs $p$ and $q$ such that $\mathcal{L}(p) = \mathcal{L}(q)$. These expressions were used to find two deterministic visibly pushdown automata

$$\mathcal{A}_1'' := (S, \Sigma_\mathbf{i}, \Sigma_\mathbf{c}, \Sigma_\mathbf{r}, \Gamma \cup \{\perp\}, \delta, I, F_1' \backslash \mathsf{IS})$$

and

$$\mathcal{A}_2'' := (S, \Sigma_\mathbf{i}, \Sigma_\mathbf{c}, \Sigma_\mathbf{r}, \Gamma \cup \{\perp\}, \delta, I, F_2' \backslash \mathsf{IS})$$

having respectively algebraic encoding

$$(\overrightarrow{I}, \mathbf{WM}, \mathbf{T_c}, \mathbf{T}_\perp, \overrightarrow{F_1' \backslash \mathsf{IS}}) \qquad \text{and} \qquad (\overrightarrow{I}, \mathbf{WM}, \mathbf{T_c}, \mathbf{T}_\perp, \overrightarrow{F_2' \backslash \mathsf{IS}})$$

such that these automata differ only in their accepting states, their set of accepting states has only accessible states, and these automata are such that

$$\begin{aligned} \vdash p &= \overrightarrow{I}^t \bullet (\mathbf{T}_\perp + \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F_1' \backslash \mathsf{IS}} \ , \\ \vdash q &= \overrightarrow{I}^t \bullet (\mathbf{T}_\perp + \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F_2' \backslash \mathsf{IS}} \ . \end{aligned}$$

So, to show

$$\vdash p = q$$

it suffices to show that

$$\vdash \overrightarrow{I}^t \bullet (\mathbf{T}_\perp + \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F_1' \backslash \mathsf{IS}} = \overrightarrow{I}^t \bullet (\mathbf{T}_\perp + \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F_2' \backslash \mathsf{IS}}.$$

In fact, we prove that $\vdash \overrightarrow{F_1' \backslash \mathsf{IS}} = \overrightarrow{F_2' \backslash \mathsf{IS}}$ and the result follows. To show $\vdash \overrightarrow{F_1' \backslash \mathsf{IS}} = \overrightarrow{F_2' \backslash \mathsf{IS}}$, the hypothesis $\mathcal{L}(p) = \mathcal{L}(q)$ is used. Since $\mathcal{A}_1''$ and $\mathcal{A}_2''$ are deterministic and they differ only in their accepting states, then the unique run of $\mathcal{A}_1''$ on any word $w$ ends in a state $s$ if and only if the unique run of $\mathcal{A}_2''$ on $w$ also ends in $s$. Since $\mathcal{L}(p) = \mathcal{L}(q)$ by hypothesis, this means that $\mathcal{L}(p)$ and $\mathcal{L}(q)$ contain the same words. So, they must have exactly the same accepting states when considering only accessible states. In other words, $F_1' \backslash \mathsf{IS} = F_2' \backslash \mathsf{IS}$. So, $\vdash \overrightarrow{F_1' \backslash \mathsf{IS}} = \overrightarrow{F_2' \backslash \mathsf{IS}}$, and then $\vdash p = q$. ■

## 4.3 Complexity of the Equational Theory of VPKA

We show the following theorem.

**Theorem 4.10** (Complexity of the equational theory of VPKA)**.** *Let* $\Sigma_{\mathbf{i}}$*,* $\Sigma_{\mathbf{c}}$ *and* $\Sigma_{\mathbf{r}}$ *be three disjoint finite sets such that at least one of them is nonempty. Let* $p$ *and* $q$ *be visibly pushdown regular expressions. The problem of deciding if* $p = q$ *using the axiomatic system of visibly pushdown Kleene algebra is EXPTIME-complete in the size of* $p$ *and* $q$*.*

*Proof.* One can see in the proof[2] of Theorem 4.2 that the "algorithm" used to decide if $\vdash p = q$ is in EXPTIME in the size of $p$ and $q$. In fact, the running time of the algorithm is dominated by the time needed for the determinization step (described in the proof of Theorem 4.8) which is in EXPTIME in the size of the automaton.

It remains to show that the problem of deciding if $p = q$ using the axiomatic system of VPKA is EXPTIME-hard. To do this, note that the proof of Theorem 3.9 shows that the problem of deciding whether two VPAs accept the same language (an EXPTIME-hard problem) can be reduced polynomially to the problem of deciding if two VPREs denote the same language. Theorems 4.1 and 4.2 show that the problem of deciding whether two VPREs denote the same language is exactly the same as deciding whether two VPREs are equivalent using only the axiomatic system of visibly pushdown Kleene algebra. So, the problem $\vdash p = q$ is hard for EXPTIME (this is the same as the language equivalence problem of visibly pushdown automata [1]). ■

It should be noted that the use of semi-visibly pushdown automata instead of plain visibly pushdown automata in the proof of Theorem 4.6 allowed us to avoid an exponential blow-up in the construction made in the proof, in particular for the case of the operator $^*$.

[2]Some parts of the proof of Theorem 4.2 are in Appendices B to E.

# Chapter 5

# Visibly Pushdown Kleene Algebra with Tests (VPKAT) and Metablocks

In order to do interprocedural program analyses using VPKA, we need a way to represent imperative interprocedural programs in VPKA. However, the representation of these programs in VPKA alone is limited. The first reason is that it lacks an essential ingredient of imperative programs: tests. The second reason is that it lacks syntactic sugar to write blocks to create clean representations of programs that look like the original programs.

This chapter addresses both issues. First, tests are added in VPKA in Section 5.1. Second, the concept of metablock is introduced in Section 5.2.

## 5.1 Visibly Pushdown Kleene Algebra with Tests

Tests are an essential ingredient to analyze imperative programs. We add them in a way similar to Kleene algebra with tests [27], presented in Section 2.1.2: a Boolean algebra $(B, +, \cdot, \overline{\ }, 0, 1)$ generated by atomic tests $\mathbf{B}$ is added to VPKA, where $B \subseteq K$. Let $\mathsf{Tests}_{\mathbf{B}}$ be the set of all test expressions that can be generated from $\mathbf{B}$ and the operators of Boolean algebra. For example, if $a, b \in \mathbf{B}$, then $b$ is a test expression of $\mathsf{Tests}_{\mathbf{B}}$ and so is $b \cdot a + \overline{a}$. Note that $0 \in \mathsf{Tests}_{\mathbf{B}}$ and $1 \in \mathsf{Tests}_{\mathbf{B}}$.

In this dissertation, we consider only Boolean algebras having a finite number of atomic elements $\mathbf{B}$. In this case, it is possible to define *atoms* of a Boolean algebra. An

atom is like an assignment of a truth value to each element of $\mathbf{B}$.

**Definition 5.1** (Atoms of a Boolean algebra)**.** Let $\mathbf{B} := \{b_1, b_2, \ldots, b_n\}$ be a finite set of atomic tests. Following Kozen [29], we define an *atom* of $\mathbf{B}$ to be any test $c_1\, c_2\, \ldots\, c_n$ such that, for all $i \in \{1, 2, \ldots, n\}$, $c_i \in \{b_i, \overline{b_i}\}$. We denote the set of all atoms of $\mathbf{B}$ by $\mathsf{Atoms}_{\mathbf{B}}$. Note that $|\mathsf{Atoms}_{\mathbf{B}}| = 2^{|\mathbf{B}|}$.

In the case that $\mathbf{B} = \emptyset$, we define the constant 1 to be the only possible atom (it is the only case in which the constant 1 is an atom). ⧫

Here is an interesting fact about atoms of a Boolean algebra. For every atom $\alpha \in \mathsf{Atoms}_{\mathbf{B}}$ and every expression (test) $b \in \mathsf{Tests}_{\mathbf{B}}$, either $\vdash \alpha \leqslant b$ or $\vdash \alpha \leqslant \overline{b}$.

We would like to use tests in operands of grammar patterns. How can we interpret a test? It seems natural to think of tests as a subset of internal actions. So, we extend the definition of explicit rewrite rules of grammar patterns to allow tests. We allow additional explicit rewrite rules of the form $P_{(x,y)} \to b$ where $b$ is a variable (or placeholder) of type $\mathsf{Tests}_{\mathbf{B}}$. Here is a more explicit definition:

**Definition 5.2** (Grammar patterns for VPKAT)**.** Let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be disjoint finite sets of atomic elements. Let $\mathbf{B}$ be a set of atomic tests. Let $V$ be a finite set of symbols (or labels) containing symbols $s$ and $t$, and let $N(V) := \{P_{(x,y)} \mid x, y \in V\}$. A *grammar pattern for* VPKAT over $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$, $\Sigma_{\mathbf{r}}$ and $\mathbf{B}$ is a partial operator represented by a tuple $G := (V, P_{(s,t)}, \to)$ where $N(V)$ is the set of nonterminals, $P_{(s,t)}$ is the starting nonterminal and $\to$ is a finite set of *explicit rewrite rules* of the form

- $P_{(x,y)} \to b$, where $b$ is a variable (or placeholder) of type $\mathsf{Tests}_{\mathbf{B}}$, and $x, y \in V$;
- $P_{(x,y)} \to a$, where $a$ is a variable (or placeholder) of type $\Sigma_{\mathbf{i}}$, and $x, y \in V$;
- $P_{(x,y)} \to c\, P_{(z,w)}\, r$, where $c$ is a variable (or placeholder) of type $\Sigma_{\mathbf{c}}$, $r$ is a variable (or placeholder) of type $\Sigma_{\mathbf{r}}$, and $w, x, y, z \in V$

and *implicit rewrite rules*

- $P_{(x,y)} \to P_{(x,z)}\, P_{(z,y)}$ for each $x, y, z \in V$.

Of course, the block notation is still possible with these grammar patterns. It suffices to consider that $[_x\, b\,]^y$ is the unary block notation for $P_{(x,y)} \to b$. ⧫

The definition of grammar patterns for VPKAT may seem surprising at first since grammar patterns for VPKA do not allow complex expressions in unary blocks. However, tests enjoy an interesting property when $\mathbf{B}$ is finite: they can be represented by a disjoint sum of atoms. So, they are easy to deal with in unary blocks as we will see in the definition of the intended model for VPKAT in Section 5.1.1.

The axioms of VPKA with tests are the axioms of Kleene algebra with tests (presented in Section 2.1.2) and axioms (3.2), (3.3), (3.4), (3.14), (3.15), (3.16) and (3.17) adapted in a natural way to allow unary blocks with tests. Furthermore, to help program analysis, we also add two axioms to VPKA with tests. To ease the reading of the new axioms, their definition uses the concept of metablock. So, we first present this concept in Section 5.2 and after we define the two new axioms for VPKA with tests in Section 5.4. Before switching to the concept of metablock, the intended model of VPKAT is presented.

### 5.1.1 Language-Theoretic Model of VPKAT

The intended model of VPKAT is a language-theoretic model that is closely linked to a trace model. In fact, it is an extension of the language-theoretic model of VPKA. The difference is that languages in VPKAT are not sets of *strings*, but sets of *guarded strings*. Guarded strings were first defined by Kaplan in 1969 [23] and reused by Kozen in 1996 to define Kleene algebra with tests [34]. As already described nicely by Kozen [29], "guarded strings are like ordinary strings over a finite alphabet $\Sigma$, except that atoms of the free Boolean algebra on a set of atomic tests $\mathbf{B}$ alternate with the symbols of $\Sigma$". Here is a more explicit definition of guarded string.

**Definition 5.3** (Guarded string [29])**.** A guarded string over alphabet $\Sigma$ and test alphabet $\mathbf{B}$ is a sequence of the form

$$\alpha_0\, p_1\, \alpha_1\, p_2\, \alpha_2\, \ldots\, \alpha_{n-1}\, p_n\, \alpha_n$$

such that $n \geqslant 0$, $\alpha_0 \in \mathsf{Atoms}_{\mathbf{B}}$ and, for every $i \in \{1, 2, \ldots, n\}$, $\alpha_i \in \mathsf{Atoms}_{\mathbf{B}}$ and $p_i \in \Sigma$. Note that, in the case $n = 0$, a guarded string is just an atom of the test alphabet. ⧫

So, a guarded string always begins and ends with an atom and the atoms alternate with symbols of $\Sigma$. Thus, "concatenation" of guarded strings cannot be the same as concatenation of languages since this would involve to have two consecutive atoms in the resulting sequence (the last atom of the first guarded string would be followed by the first atom of the second guarded string), and so, it would violate the definition

of guarded string. The previous reasoning is just a "syntactic" problem, but it hides a "semantic" problem. Recall that we said that we could consider an atom like an assignment of a truth value to each element of $\mathbf{B}$. So, the meaning of two consecutive atoms is like potentially assigning two different truth values to an element of $\mathbf{B}$ *at the same "time"*![1] This should not happen in this framework unless the two atoms are identical (they share the same truth value for each element of $\mathbf{B}$). The "concatenation" operation deriving from this idea is called the *coalesced product* for guarded strings. Suppose $q\,\alpha$ is a guarded string ending with the atom $\alpha$ and suppose that $\beta\,r$ is a guarded string beginning with the atom $\beta$. The coalesced product $\diamond$ of these two guarded strings is defined by

$$q\,\alpha \diamond \beta\,r := \begin{cases} q\,\alpha\,r & \text{if } \alpha = \beta \ , \\ \text{undefined} & \text{otherwise} \ . \end{cases}$$

So, the coalesced product of two guarded strings merges the last atom of the first guarded string with the first atom of the second guarded string *if the two atoms are the same*. Otherwise, the coalesced product is undefined. So, this operation is a partial operation when applied to guarded strings. However, it is a *total* operation when applied to *sets* of guarded strings. Let $S$ and $T$ be sets of guarded strings. The coalesced product of these two sets is defined by

$$S \diamond T := \{s \diamond t \mid s \in S \wedge t \in T\} \ .$$

The coalesced product plays the same role for guarded strings as the concatenation operation plays for ordinary strings. However, this implies that the unit of the concatenation (the set only containing the empty word) should also be replaced for the coalesced product. In the language-theoretic model of guarded strings, the set $\{\varepsilon\}$ (playing the role of the unit of the concatenation) is now replaced by the set of all atoms $\mathsf{Atoms}_{\mathbf{B}}$. Besides, the Kleene star operation on guarded strings is defined to use the coalesced product instead of the concatenation operation. Thus, for every set $S$ of guarded strings, define $S^0 := \mathsf{Atoms}_{\mathbf{B}}$ and $S^{n+1} := S \diamond S^n$ for every $n \in \mathbb{N}$. Then,

$$S^* := (\cup\, n \mid n \in \mathbb{N} : S^n) \ .$$

It is already known that the powerset of all guarded strings over an alphabet $\Sigma$ and a test alphabet $\mathbf{B}$ forms a Kleene algebra with binary operations $\cup$ and $\diamond$, unary operation $^*$ and constants $\mathsf{Atoms}_{\mathbf{B}}$ and $\emptyset$ [34]. Also, the powerset of all atoms $\mathsf{Atoms}_{\mathbf{B}}$ forms a

[1] "Time" in this framework is more like a specific *state* in a computation. In this settings, atoms denote a state and elements of $\Sigma$ (though of as instructions of a program) allow one to change the current state.

Boolean algebra with binary operations $\cup$ and $\diamond$, unary complementation operation $\overline{\phantom{x}}$ (defined, for every $A \subseteq \mathsf{Atoms}_{\mathbf{B}}$, by $\overline{A} := \mathsf{Atoms}_{\mathbf{B}} - A$) and constants $\mathsf{Atoms}_{\mathbf{B}}$ and $\emptyset$. Thus, the two algebras combined give a Kleene algebra with tests. Also, the standard interpretation $\mathsf{G}$ for this algebra is defined, for every atomic test $b \in \mathbf{B}$ and every atomic element $p \in \Sigma$, by

$$\mathsf{G}(0) := \emptyset, \qquad \mathsf{G}(1) := \mathsf{Atoms}_{\mathbf{B}}, \qquad \mathsf{G}(p) := \{\alpha\, p\, \beta \mid \alpha, \beta \in \mathsf{Atoms}_{\mathbf{B}}\},$$
$$\mathsf{G}(b) := \{\alpha \mid \alpha \in \mathsf{Atoms}_{\mathbf{B}} \wedge b \text{ occurs positively in } \alpha\},$$

and extends over the structure of regular expressions with tests with $\cdot$ becoming $\diamond$, $+$ becoming $\cup$, $^*$ becoming the set operator $^*$ on guarded strings, and $\overline{\phantom{x}}$ becoming the complementation operator $\overline{\phantom{x}}$ on the powerset of $\mathsf{Atoms}_{\mathbf{B}}$.

We will now see that this model of Kleene algebra with tests can be extended to form a model of visibly pushdown Kleene algebra with tests. In this case, the alphabet $\Sigma$ is simply $\Sigma_{\mathbf{i}} \cup \Sigma_{\mathbf{c}} \cup \Sigma_{\mathbf{r}}$. Besides, the infinite family of operators must be defined for guarded strings. To do this, we introduce the concept of a well-matched visibly pushdown grammar on guarded strings.

**Definition 5.4** (Well-matched visibly pushdown grammar (WMVPG) on guarded strings)**.** Let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be disjoint finite sets of atomic elements. Let $\mathbf{B}$ be a set of atomic tests. Let $V$ be a finite set of symbols (or labels) containing symbols $s$ and $t$, and let $N(V) := \{P_{(x,y)} \mid x, y \in V\}$. A *well-matched visibly pushdown grammar on guarded strings* over $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$, $\Sigma_{\mathbf{r}}$ and $\mathbf{B}$ is a tuple $G := (V, P_{(s,t)}, \rightarrow)$ where $N(V)$ is the set of nonterminals, $P_{(s,t)}$ is the starting nonterminal and $\rightarrow$ is a finite set of *explicit rewrite rules* of the form

- $P_{(x,y)} \rightarrow \alpha$, where $\alpha \in \mathsf{Atoms}_{\mathbf{B}}$ and $x, y \in V$;
- $P_{(x,y)} \rightarrow \alpha_0\, a\, \alpha_1$, where $\alpha_0, \alpha_1 \in \mathsf{Atoms}_{\mathbf{B}}$, $a \in \Sigma_{\mathbf{i}}$ and $x, y \in V$;
- $P_{(x,y)} \rightarrow \alpha_0\, c\, \alpha_1 \diamond P_{(z,w)} \diamond \alpha_2\, r\, \alpha_3$, where $\alpha_0, \alpha_1, \alpha_2, \alpha_3 \in \mathsf{Atoms}_{\mathbf{B}}$, $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $w, x, y, z \in V$

and *implicit rewrite rules*

- $P_{(x,y)} \rightarrow P_{(x,z)} \diamond P_{(z,y)}$ for each $x, y, z \in V$. ♦

Note that WMVPGs on guarded strings are different from usual context-free grammars. First, the terminals of the grammar are guarded strings instead of strings. Second, the "concatenation" operation used between elements of the right-hand side of a

rule is the coalesced product. This use of the coalesced product has a slight impact on the definition of "words" generated by such a grammar. After all, the coalesced product of guarded strings may be undefined in some cases. To handle this case, the idea of the coalesced product on sets of guarded strings is used: the words generated by a WMVPG on guarded strings is a set containing only the correctly defined guarded strings that can be generated from the grammar. Note that each guarded string in this set is finite.

As for WMVPGs, a "block" notation for WMVPGs on guarded strings can be defined. Let $G := (V, P_{(s,t)}, \to)$ be a WMVPG. We write

$$\underset{x}{[}\, \alpha \,\overset{y}{]}, \qquad \underset{x}{[}\, \alpha_0\, a\, \alpha_1 \,\overset{y}{]}, \qquad \underset{x}{[}\, \alpha_0\, c\, \alpha_1 \underset{z}{\downarrow} \overset{w}{\uparrow} \alpha_2\, r\, \alpha_3 \,\overset{y}{]}$$

to respectively represent the explicit rewrite rules

$$P_{(x,y)} \to \alpha, \qquad P_{(x,y)} \to \alpha_0\, a\, \alpha_1, \qquad P_{(x,y)} \to \alpha_0\, c\, \alpha_1 \diamond P_{(z,w)} \diamond \alpha_2\, r\, \alpha_3 \ .$$

We call *unary block* each rule of the form $[_x\, \alpha\, ]^y$ or $[_x\, \alpha_0\, a\, \alpha_1\, ]^y$. We also call *binary block* each rule of the form $[_x\, \alpha_0\, c\, \alpha_1 \downarrow_z \uparrow^w \alpha_2\, r\, \alpha_3\, ]^y$.

Let $\mathcal{B}$ be the block notation of the explicit rewrite rules of $G$. Then, $G$ is abbreviated as $(\!|_s\, \mathcal{B}\, |\!)^t$.

Here are some examples of languages generated by WMVPGs on guarded strings, using the block notation. Let $\Sigma_{\mathbf{i}} := \emptyset$, $\Sigma_{\mathbf{c}} := \{\mathsf{c}\}$, $\Sigma_{\mathbf{r}} := \{\mathsf{r}\}$, $\mathbf{B} := \{\mathsf{b}\}$ and $V := \{\mathsf{x}, \mathsf{y}\}$. Then,

- $(\!|\, \underset{\mathsf{x}\,\mathsf{x}}{[}\, \mathsf{b}\,\mathsf{c}\,\mathsf{b} \underset{\mathsf{x}}{\downarrow} \overset{\mathsf{y}}{\uparrow} \mathsf{b}\,\mathsf{r}\,\mathsf{b} \,\overset{\mathsf{y}}{]}, \underset{\mathsf{x}}{[}\, \mathsf{b} \,\overset{\mathsf{y}\,\mathsf{y}}{]}\, |\!) = \{(\mathsf{b}\,\mathsf{c})^n\, \mathsf{b}\, (\mathsf{r}\,\mathsf{b})^n \mid n \in \mathbb{N}\}$,
- $(\!|\, \underset{\mathsf{x}\,\mathsf{x}}{[}\, \mathsf{b}\,\mathsf{c}\,\overline{\mathsf{b}} \underset{\mathsf{x}}{\downarrow} \overset{\mathsf{y}}{\uparrow} \overline{\mathsf{b}}\,\mathsf{r}\,\mathsf{b} \,\overset{\mathsf{y}}{]}, \underset{\mathsf{x}}{[}\, \mathsf{b} \,\overset{\mathsf{y}\,\mathsf{y}}{]}\, |\!) = \{\mathsf{b}\}$.

Note that, in the second example, the binary block cannot be used to generate a correctly defined guarded string since the guarded string $\mathsf{b}\,\mathsf{c}\,\overline{\mathsf{b}}$ ends with the atom $\overline{\mathsf{b}}$ and since no block of the grammar begins with the atom $\overline{\mathsf{b}}$.

The set operators on guarded strings are now all defined. To complete the definition of the model, a link (an *interpretation*) between operators and atomic elements of VPKAT, and the set operators on guarded strings and atomic elements of guarded strings must be defined. To do this, it suffices to extend properly the standard interpretation $\mathsf{G}$ for Kleene algebra with tests to every expression of visibly pushdown Kleene algebra

with tests. Thus, the standard interpretation $\mathsf{G}$ for $\mathsf{VPKAT}$ is defined, for every atomic test $b \in \mathbf{B}$, every internal action $a \in \Sigma_{\mathbf{i}}$, every call action $c \in \Sigma_{\mathbf{c}}$ and every return action $r \in \Sigma_{\mathbf{r}}$, by

$$\mathsf{G}(0) := \emptyset, \qquad \mathsf{G}(1) := \mathsf{Atoms}_{\mathbf{B}}, \qquad \mathsf{G}(a) := \{\alpha\, a\, \beta \mid \alpha, \beta \in \mathsf{Atoms}_{\mathbf{B}}\},$$
$$\mathsf{G}(c) := \{\alpha\, c\, \beta \mid \alpha, \beta \in \mathsf{Atoms}_{\mathbf{B}}\}, \qquad \mathsf{G}(r) := \{\alpha\, r\, \beta \mid \alpha, \beta \in \mathsf{Atoms}_{\mathbf{B}}\},$$
$$\mathsf{G}(b) := \{\alpha \mid \alpha \in \mathsf{Atoms}_{\mathbf{B}} \wedge b \text{ occurs positively in } \alpha\},$$

and extends over the structure of visibly pushdown regular expressions with $\cdot$ becoming $\diamond$, $+$ becoming $\cup$, $^*$ becoming the set operator $^*$ on guarded strings, and $\overline{\phantom{x}}$ becoming the complementation operator $\overline{\phantom{x}}$ on the powerset of $\mathsf{Atoms}_{\mathbf{B}}$. Also, a grammar pattern for $\mathsf{VPKAT}$ *along with its operands* (note this expression $G$) becomes (is interpreted as) a $\mathsf{WMVPG}$ on guarded strings in the following way:

- every explicit rewrite rule of the form $P_{(x,y)} \to b$ in which $b \in \mathsf{Tests}_{\mathbf{B}}$ in $G$ becomes the following rewrite rules for a $\mathsf{WMVPG}$ on guarded strings:

$$P_{(x,y)} \to \alpha, \qquad \text{for all } \alpha \in \mathsf{Atoms}_{\mathbf{B}} \text{ such that } \alpha \leqslant b \ ,$$

- every explicit rewrite rule of the form $P_{(x,y)} \to a$ in which $a \in \Sigma_{\mathbf{i}}$ in $G$ becomes the following rewrite rules for a $\mathsf{WMVPG}$ on guarded strings:

$$P_{(x,y)} \to \alpha\, a\, \beta, \qquad \text{for all } \alpha, \beta \in \mathsf{Atoms}_{\mathbf{B}} \ ,$$

- every explicit rewrite rule of the form $P_{(x,y)} \to c\, P_{(z,w)}\, r$ in which $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$ in $G$ becomes the following rewrite rules for a $\mathsf{WMVPG}$ on guarded strings:

$$P_{(x,y)} \to \alpha_0\, c\, \alpha_1 \diamond P_{(z,w)} \diamond \alpha_2\, r\, \alpha_3, \qquad \text{for all } \alpha_0, \alpha_1, \alpha_2, \alpha_3 \in \mathsf{Atoms}_{\mathbf{B}} \ .$$

So, $\mathsf{G}(G)$ is the language generated by the $\mathsf{WMVPG}$ on guarded strings represented by $G$.

It is routine to prove that the powerset of all guarded strings along with its set operators is a model of visibly pushdown Kleene algebra with tests.

### Alternative Definition of the Family of Operators for Guarded Strings

Like for the definition of $\mathcal{G}$ for $\mathsf{VPKA}$, there exists an alternative definition of the preceding infinite family of operators that uses directly the notion of "block" instead of using the detour of grammars. The idea behind a block expression $(\!|_x\, \mathcal{B}\, |\!)^{x'}$ is to generate

any well-matched guarded string that can be produced by a correct "travelling" of the list of blocks, starting the travel in any block that has $x$ as starting label and ending it in any block that has $x'$ as ending label. In essence, a correct travelling of a $(\!|\,|\!)$-expression is similar to the following derivation strategy for a WMVPG on guarded strings: always use an explicit rewrite rule to derive the *first* nonterminal generated by an implicit rewrite rule. Such correct travelling is easy to infer from the definition of a correct travelling of Section 3.1.2 and the definition of blocks for guarded strings.

Let $\mathcal{B}$ be a finite list of unary and binary blocks on a finite set of labels $V$. Define $\mathcal{B}^1$ as the set of unary blocks of $\mathcal{B}$ and $\mathcal{B}^2$ as the set of binary blocks of $\mathcal{B}$. Note that, in the following, we denote a unary block by $[_x\, m\,]^y \in \mathcal{B}^1$ for labels $x$ and $y$, but $m$ can be either an atom $\alpha \in \mathsf{Atoms_B}$ or an internal action surrounded by atoms $\alpha\, a\, \beta$ where $\alpha, \beta \in \mathsf{Atoms_B}$ and $a \in \Sigma_\mathbf{i}$. This is to facilitate the reading of the expression.

For $n \in \mathbb{N}$, define the *power-recursion operator on guarded strings* $(\!|_x\, \mathcal{B}\, |\!)^y_n$, where $x, y \in V$, by induction on $n$: $(\!|_x\, \mathcal{B}\, |\!)^y_0 := (\cup\, m \mid [_x\, m\,]^y \in \mathcal{B}^1 : \{m\})$, and

$$
\begin{aligned}
(\!|\, \mathcal{B} \,|\!)^{\,y}_{x\;\;n+1} :=\; & (\cup\, m, v \mid [_x\, m\,]^v \in \mathcal{B}^1 : \{m\} \diamond (\!|\, \mathcal{B} \,|\!)^{\,y}_{v\;\;n}) \\
& \cup\; (\cup\, \alpha_0, c, \alpha_1, z, \alpha_2, r, \alpha_3, w \mid [_x\, \alpha_0\, c\, \alpha_1 \downarrow^w_z \uparrow \alpha_2\, r\, \alpha_3,\,]^y \in \mathcal{B}^2 : \\
& \qquad \{\alpha_0\, c\, \alpha_1\} \diamond (\!|\, \mathcal{B} \,|\!)^{\,w}_{z\;\;n} \diamond \{\alpha_2\, r\, \alpha_3\}) \\
& \cup\; (\cup\, \alpha_0, c, \alpha_1, z, \alpha_2, r, \alpha_3, w, v, n_1, n_2 \mid [_x\, \alpha_0\, c\, \alpha_1 \downarrow^w_z \uparrow \alpha_2\, r\, \alpha_3\,]^v \in \mathcal{B}^2 \\
& \qquad \wedge\; n_1, n_2 \in \mathbb{N} \wedge n_1 + n_2 = n - 1 : \\
& \qquad \{\alpha_0\, c\, \alpha_1\} \diamond (\!|\, \mathcal{B} \,|\!)^{\,w}_{z\;\;n_1} \diamond \{\alpha_2\, r\, \alpha_3\} \diamond (\!|\, \mathcal{B} \,|\!)^{\,y}_{v\;\;n_2}) \;\; .
\end{aligned}
$$

Intuitively, $(\!|_x\, \mathcal{B}\, |\!)^y_n$ denotes the set of all well-matched guarded strings that can be generated by any correct travelling of $\mathcal{B}$ of length $n+1$ starting with $x$ and ending with $y$. With this definition, it is easy to define an operator $(\!|_x\, \mathcal{B}\, |\!)^y$ by

$$
(\!|\, \mathcal{B} \,|\!)^{\,y}_{x} := (\cup\, n \mid n \in \mathbb{N} : (\!|\, \mathcal{B} \,|\!)^{\,y}_{x\;\;n}) \;\; .
$$

Like for the block-based definition of the infinite family of operators in Section 3.1.2, it is possible to define a notion of *labelled block pattern for* VPKAT that is similar to the notion of grammar patterns for VPKAT.

Accordingly, the interpretation of these labelled block patterns for VPKAT in the model of guarded strings is also similar to the interpretation of grammar patterns for

VPKAT in the model of guarded strings. In particular, a labelled block pattern for VPKAT *along with its operands* (note this expression $(\!|_x \mathcal{B} |\!)^y$) becomes (is interpreted as) a block expression on guarded strings $(\!|_x \mathsf{G_B}(\mathcal{B}) |\!)^y$ where $\mathsf{G_B}(\mathcal{B})$ is a function applied to each block of $\mathcal{B}$ that gives a list of blocks on guarded strings in the following way:

- for every unary block of the form $[\!]_z\, b\,]\!]^w$ in which $b \in \mathsf{Tests_B}$ in $\mathcal{B}^1$, $\mathsf{G_B}([\!]_z\, b\,]\!]^w)$ becomes a list of blocks on guarded strings containing $[\!]_z\, \alpha\,]\!]^w$ for each $\alpha \in \mathsf{Atoms_B}$ such that $\alpha \leqslant b$,
- for every unary block of the form $[\!]_z\, a\,]\!]^w$ in which $a \in \Sigma_\mathbf{i}$ in $\mathcal{B}^1$, $\mathsf{G_B}([\!]_z\, a\,]\!]^w)$ becomes a list of blocks on guarded strings containing $[\!]_z\, \alpha\, a\, \beta\,]\!]^w$ for each $\alpha, \beta \in \mathsf{Atoms_B}$,
- for every binary block of the form $[\!]_z\, c{\downarrow_v}{\uparrow^{v'}}\, r\,]\!]^w$ in $\mathcal{B}^2$, $\mathsf{G_B}([\!]_z\, c{\downarrow_v}{\uparrow^{v'}}\, r\,]\!]^w)$ becomes a list of blocks on guarded strings containing $[\!]_z\, \alpha_0\, c\, \alpha_1 {\downarrow_v}{\uparrow^{v'}}\, \alpha_2\, r\, \alpha_3\,]\!]^w$ for each $\alpha_0, \alpha_1, \alpha_2, \alpha_3 \in \mathsf{Atoms_B}$.

## 5.2 Metablocks

The forms of explicit rewrite rules (the blocks in $(\!|\,|\!)$-expressions) for grammar patterns for VPKAT are simple, but it can be tedious to write such rewrite rules for a large expression. To simplify this process, we define *metablocks*, which are abbreviations (similar to regular expressions) of a list of blocks. Thus, metablocks allow us to write more complex explicit rewrite rules to ease program manipulation.

Let $\Sigma_\mathbf{i}$, $\Sigma_\mathbf{c}$, $\Sigma_\mathbf{r}$ and $\mathbf{B}$ be finite sets. Let $V$ be a finite set of symbols. A metablock is an expression $[\![_x\, e\,]\!]^y$ where $x, y \in V$ and $e$ is an expression of the set $\mathsf{MBexp}$ that is defined as the smallest set containing $a$ for each $a \in \Sigma_\mathbf{i} \cup \mathsf{Tests_B}$, $(c{\downarrow_z}{\uparrow^w}\, r)$ for each $c \in \Sigma_\mathbf{c}$, $r \in \Sigma_\mathbf{r}$ and $z, w \in V$, and closed under "operators" $\cdot$, $+$ and $^*$. Note that metablocks are written with bolder square brackets than blocks. A metablock is reduced to a list of unary and binary blocks by the function $\mathsf{mb}$ defined inductively by:

- $\mathsf{mb}([\![_x\, a\,]\!]^y) := [\!]_x\, a\,]\!]^y$ for $a \in \Sigma_\mathbf{i} \cup \mathsf{Tests_B}$ and $x, y \in V$;
- $\mathsf{mb}([\![_x (c{\downarrow_z}{\uparrow^w}\, r)\,]\!]^y) := [\!]_x\, c{\downarrow_z}{\uparrow^w}\, r\,]\!]^y$ for $c \in \Sigma_\mathbf{c}$, $r \in \Sigma_\mathbf{r}$ and $x, y, z, w \in V$;
- $\mathsf{mb}([\![_x\, p \cdot q\,]\!]^y) := \mathsf{mb}([\![_x\, p\,]\!]^z), \mathsf{mb}([\![_z\, q\,]\!]^y)$ for $x, y \in V$ and a fresh label $z$ (a label not in $V$);
- $\mathsf{mb}([\![_x\, p + q\,]\!]^y) := \mathsf{mb}([\![_x\, p\,]\!]^y), \mathsf{mb}([\![_x\, q\,]\!]^y)$ for $x, y \in V$;

- $\mathsf{mb}(\llbracket_x\, p^*\, \rrbracket^y) := \llbracket_x\, 1\, \rrbracket^y, \mathsf{mb}(\llbracket_x\, p\, \rrbracket^y), \mathsf{mb}(\llbracket_x\, p\, \rrbracket^z), \mathsf{mb}(\llbracket_z\, p\, \rrbracket^z), \mathsf{mb}(\llbracket_z\, p\, \rrbracket^y)$ for $x, y \in V$ and a fresh label $z$.

Note that, in the following sections, we allow ourselves to apply the function $\mathsf{mb}$ to lists of metablocks instead of a single metablock.

To simplify metablock expressions, we write $pq$ instead of $p \cdot q$. However, this abbreviation is ambiguous for tests since, for $a, b \in \mathbf{B}$, the metablock $\llbracket_x\, ab\, \rrbracket^y$ both represents the unary block $\llbracket_x\, a \cdot b\, \rrbracket^y$ and the metablock $\llbracket_x\, a \cdot b\, \rrbracket^y$. To avoid this ambiguity, we consider that $\llbracket_x\, ab\, \rrbracket^y$ always represents the unary block $\llbracket_x\, a \cdot b\, \rrbracket^y$. This extra syntactic care is not really useful since we will see in the following sections that the unary block $\llbracket_x\, a \cdot b\, \rrbracket^y$ and the metablock $\llbracket_x\, a \cdot b\, \rrbracket^y$ behave similarly (as one should expect).

Here are some examples of metablocks and their reduction to lists of unary and binary blocks. Let $\Sigma_{\mathbf{i}} := \{\mathsf{a}_1, \mathsf{a}_2\}$, $\Sigma_{\mathbf{c}} := \{\mathsf{c}\}$, $\Sigma_{\mathbf{r}} := \{\mathsf{r}\}$, $\mathbf{B} := \{\mathsf{b}\}$ and $V := \{\mathsf{x}, \mathsf{y}\}$. The fresh labels $\mathsf{t}_1$, $\mathsf{t}_2$, $\mathsf{t}_3$, $\mathsf{t}_4$, $\mathsf{t}_5$, $\mathsf{v}_1$, $\mathsf{v}_2$, $\mathsf{v}_3$, $\mathsf{v}_4$, $\mathsf{v}$, $\mathsf{w}$ and $\mathsf{z}$ will also be used. Suppose that every following example is independent from the others, so the fresh labels are always fresh at the beginning of a new example. Then,

- $\mathsf{mb}(\llbracket_{\mathsf{x}}\, \mathsf{b} \cdot \mathsf{a}_1 + \overline{\mathsf{b}} \cdot \mathsf{a}_2\, \rrbracket^{\mathsf{y}}) =$
  $\llbracket_{\mathsf{x}}\, \mathsf{b}\, \rrbracket^{\mathsf{z}},\ \llbracket_{\mathsf{z}}\, \mathsf{a}_1\, \rrbracket^{\mathsf{y}},\ \llbracket_{\mathsf{x}}\, \overline{\mathsf{b}}\, \rrbracket^{\mathsf{w}},\ \llbracket_{\mathsf{w}}\, \mathsf{a}_2\, \rrbracket^{\mathsf{y}}$ ;
- $\mathsf{mb}\big(\llbracket_{\mathsf{x}}(\mathsf{b} \cdot \mathsf{a}_1 + \overline{\mathsf{b}} \cdot \mathsf{a}_2)^*\, \rrbracket^{\mathsf{y}}\big) =$
  $\llbracket_{\mathsf{x}}\, 1\, \rrbracket^{\mathsf{y}},\ \llbracket_{\mathsf{x}}\, \mathsf{b}\, \rrbracket^{\mathsf{v}_1},\ \llbracket_{\mathsf{v}_1}\, \mathsf{a}_1\, \rrbracket^{\mathsf{y}},\ \llbracket_{\mathsf{x}}\, \overline{\mathsf{b}}\, \rrbracket^{\mathsf{t}_1},\ \llbracket_{\mathsf{t}_1}\, \mathsf{a}_2\, \rrbracket^{\mathsf{y}},\ \llbracket_{\mathsf{x}}\, \mathsf{b}\, \rrbracket^{\mathsf{v}_2},\ \llbracket_{\mathsf{v}_2}\, \mathsf{a}_1\, \rrbracket^{\mathsf{z}},\ \llbracket_{\mathsf{x}}\, \overline{\mathsf{b}}\, \rrbracket^{\mathsf{t}_2},\ \llbracket_{\mathsf{t}_2}\, \mathsf{a}_2\, \rrbracket^{\mathsf{z}},$
  $\llbracket_{\mathsf{z}}\, \mathsf{b}\, \rrbracket^{\mathsf{v}_3},\ \llbracket_{\mathsf{v}_3}\, \mathsf{a}_1\, \rrbracket^{\mathsf{z}},\ \llbracket_{\mathsf{z}}\, \overline{\mathsf{b}}\, \rrbracket^{\mathsf{t}_3},\ \llbracket_{\mathsf{t}_3}\, \mathsf{a}_2\, \rrbracket^{\mathsf{z}},\ \llbracket_{\mathsf{z}}\, \mathsf{b}\, \rrbracket^{\mathsf{v}_4},\ \llbracket_{\mathsf{v}_4}\, \mathsf{a}_1\, \rrbracket^{\mathsf{y}},\ \llbracket_{\mathsf{z}}\, \overline{\mathsf{b}}\, \rrbracket^{\mathsf{t}_4},\ \llbracket_{\mathsf{t}_4}\, \mathsf{a}_2\, \rrbracket^{\mathsf{y}}$ ;
- $\mathsf{mb}\Big(\llbracket_{\mathsf{x}}\, \mathsf{a}_1 \cdot \big(\mathsf{b} \cdot (\mathsf{c}{\downarrow_{\mathsf{x}}}{\uparrow^{\mathsf{y}}}\mathsf{r})\big)^* \cdot \overline{\mathsf{b}} \cdot \mathsf{a}_2\, \rrbracket^{\mathsf{y}}\Big) =$
  $\llbracket_{\mathsf{x}}\, \mathsf{a}_1\, \rrbracket^{\mathsf{z}},\ \llbracket_{\mathsf{z}}\, 1\, \rrbracket^{\mathsf{w}},\ \llbracket_{\mathsf{z}}\, \mathsf{b}\, \rrbracket^{\mathsf{t}_1},\ \llbracket_{\mathsf{t}_1}\, \mathsf{c}{\downarrow_{\mathsf{x}}}{\uparrow^{\mathsf{y}}}\mathsf{r}\, \rrbracket^{\mathsf{w}},\ \llbracket_{\mathsf{z}}\, \mathsf{b}\, \rrbracket^{\mathsf{t}_2},\ \llbracket_{\mathsf{t}_2}\, \mathsf{c}{\downarrow_{\mathsf{x}}}{\uparrow^{\mathsf{y}}}\mathsf{r}\, \rrbracket^{\mathsf{v}},$
  $\llbracket_{\mathsf{v}}\, \mathsf{b}\, \rrbracket^{\mathsf{t}_3},\ \llbracket_{\mathsf{t}_3}\, \mathsf{c}{\downarrow_{\mathsf{x}}}{\uparrow^{\mathsf{y}}}\mathsf{r}\, \rrbracket^{\mathsf{v}},\ \llbracket_{\mathsf{v}}\, \mathsf{b}\, \rrbracket^{\mathsf{t}_4},\ \llbracket_{\mathsf{t}_4}\, \mathsf{c}{\downarrow_{\mathsf{x}}}{\uparrow^{\mathsf{y}}}\mathsf{r}\, \rrbracket^{\mathsf{w}},\ \llbracket_{\mathsf{w}}\, \overline{\mathsf{b}}\, \rrbracket^{\mathsf{t}_5},\ \llbracket_{\mathsf{t}_5}\, \mathsf{a}_2\, \rrbracket^{\mathsf{y}}$ .

It is clear from the previous examples that metablocks ease the comprehension of a list of unary and binary blocks. Note that this is possible by mimicking the standard operators of regular expressions in a metablock.

So, we now have an interesting tool to ease the comprehension of (or to model) a list of unary and binary blocks: metablocks. However, *modelling* is not enough. We should also provide *reasoning tools* that directly use the notion of metablock. This way, proofs in VPKAT should be easier to do. This is the goal of the following section.

## 5.3 Extension of Some Laws to Metablocks

It is easy to see that some simple laws of $(\!|\,|\!)$-expressions carry over to $(\!|\,|\!)$-expressions containing metablocks. Axiom (3.4), the law of idempotency of blocks, the law of commutativity of blocks and the law of the removal of unused blocks are such examples. However, there are some laws of $(\!|\,|\!)$-expressions that are difficult to extend to metablocks. We discuss such laws here (these laws first appeared in [11]).

To facilitate the discussion, let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be disjoint finite sets of atomic elements, $\mathbf{B}$ be a set of atomic tests and $V$ be a finite set of labels. Let $\mathcal{B}$ be a finite list of *metablocks* on a finite set of labels $V$. Let $\mathcal{B}^{\mathsf{set}}$ be the set containing every metablock of $\mathcal{B}$. Let $\theta$ be a substitution environment (a partial function) taking a pair of labels from $V$ and returning a visibly pushdown regular expression.

We define a useful function for metablocks. Let $\mathsf{mb\_vpre}_\theta$ be a function (relative to $\theta$), that creates a visibly pushdown regular expression from an expression of $\mathsf{MBexp}$, defined inductively by:

- $\mathsf{mb\_vpre}_\theta(a) := a$, for $a \in \Sigma_{\mathbf{i}} \cup \mathsf{Tests}_{\mathbf{B}}$;
- $\mathsf{mb\_vpre}_\theta((c\!\downarrow_z \uparrow^w r)) := c \cdot \theta(z,w) \cdot r$, for $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $z, w \in V$;
- $\mathsf{mb\_vpre}_\theta(p \cdot q) := \mathsf{mb\_vpre}_\theta(p) \cdot \mathsf{mb\_vpre}_\theta(q)$;
- $\mathsf{mb\_vpre}_\theta(p + q) := \mathsf{mb\_vpre}_\theta(p) + \mathsf{mb\_vpre}_\theta(q)$;
- $\mathsf{mb\_vpre}_\theta(p^*) := (\mathsf{mb\_vpre}_\theta(p))^*$.

Here are some theorems involving $\mathsf{mb\_vpre}_\theta$. Let $\theta_{\mathcal{B}}$ be the substitution environment defined by, for all $u, u' \in V$,

$$\theta_{\mathcal{B}}(u,u') := \underset{u}{(\!|\,\mathcal{B}\,|\!)}^{u'} \quad .$$

Then, a generalization of the unfolding axioms (3.2) and (3.3) can be proved:

$$\mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(m) \leqslant \underset{x}{(\!|\,\mathcal{B}\,|\!)}^{y} \quad , \qquad \text{if } \underset{x}{[\,m\,]}^{y} \in \mathcal{B}^{\mathsf{set}} \quad . \tag{5.1}$$

*Proof.* We prove (5.1) by structural induction on $m$. For the base case where $m \in \Sigma_{\mathbf{i}} \cup \mathsf{Tests}_{\mathbf{B}}$, the proof is direct by the definition of metablock, the fact that $\mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(m) = m$ and axiom (3.2).

For the base case where $m$ is an expression $(c{\downarrow_z}{\uparrow^w} r)$, the proof is direct by the definition of metablock, the fact that $\mathsf{mb\_vpre}_{\theta_\mathcal{B}}((c{\downarrow_z}{\uparrow^w} r)) = c \cdot (\!|_z \mathcal{B} |\!)^w \cdot r$ and axiom (3.3).

For the inductive case where $m = p \cdot q$, we first consider the metablock $[\!|_x p \cdot q |\!]^y$ in $\mathcal{B}$. We suppose that $u$ is the fresh label used by $\mathsf{mb}$. In other words, we suppose that

$$\mathop{\mathsf{mb}}_{x}([\!| p \cdot q |\!]^{y}) \quad = \quad \mathop{\mathsf{mb}}_{x}([\!| p |\!]^{u}),\ \mathop{\mathsf{mb}}_{u}([\!| q |\!]^{y}) \ .$$

We also suppose that $\mathsf{mb\_vpre}_{\theta_\mathcal{B}}(p) \leqslant (\!|_x \mathcal{B} |\!)^u$ and $\mathsf{mb\_vpre}_{\theta_\mathcal{B}}(q) \leqslant (\!|_u \mathcal{B} |\!)^y$. By the definition of $\mathsf{mb\_vpre}_{\theta_\mathcal{B}}$, we have to prove that

$$\mathsf{mb\_vpre}_{\theta_\mathcal{B}}(p) \cdot \mathsf{mb\_vpre}_{\theta_\mathcal{B}}(q) \leqslant \mathop{(\!|}_{x} \mathcal{B} \mathop{|\!)}^{y} \ .$$

By the two previous inductive hypotheses and Kleene algebra, it suffices to prove that

$$\mathop{(\!|}_{x} \mathcal{B} \mathop{|\!)}^{u} \cdot \mathop{(\!|}_{u} \mathcal{B} \mathop{|\!)}^{y} \leqslant \mathop{(\!|}_{x} \mathcal{B} \mathop{|\!)}^{y} \ .$$

This is direct by axiom (3.4) extended to metablocks.

For the inductive case where $m = p + q$, by the definition of $\mathsf{mb\_vpre}_{\theta_\mathcal{B}}$, we suppose that $\mathsf{mb\_vpre}_{\theta_\mathcal{B}}(p) \leqslant (\!|_x \mathcal{B} |\!)^y$ and $\mathsf{mb\_vpre}_{\theta_\mathcal{B}}(q) \leqslant (\!|_x \mathcal{B} |\!)^y$. We have to prove that

$$\mathsf{mb\_vpre}_{\theta_\mathcal{B}}(p) + \mathsf{mb\_vpre}_{\theta_\mathcal{B}}(q) \leqslant \mathop{(\!|}_{x} \mathcal{B} \mathop{|\!)}^{y} \ .$$

By the two previous inductive hypotheses and Kleene algebra, it suffices to prove that

$$\mathop{(\!|}_{x} \mathcal{B} \mathop{|\!)}^{y} + \mathop{(\!|}_{x} \mathcal{B} \mathop{|\!)}^{y} \leqslant \mathop{(\!|}_{x} \mathcal{B} \mathop{|\!)}^{y} \ .$$

This is direct by the idempotency of $+$.

For the inductive case where $m = p^*$, we first consider the metablock $[\!|_x p^* |\!]^y$ in $\mathcal{B}$. We suppose that $u$ is the fresh label used by $\mathsf{mb}$. In other words, we suppose that

$$\mathop{\mathsf{mb}}_{x}([\!| p^* |\!]^{y}) \quad = \quad \mathop{[\,1\,]}_{x}^{y},\ \mathop{\mathsf{mb}}_{x}([\!| p |\!]^{y}),\ \mathop{\mathsf{mb}}_{x}([\!| p |\!]^{u}),\ \mathop{\mathsf{mb}}_{u}([\!| p |\!]^{u}),\ \mathop{\mathsf{mb}}_{u}([\!| p |\!]^{y}) \ .$$

We also suppose that

$$\mathsf{mb\_vpre}_{\theta_\mathcal{B}}(p) \leqslant \mathop{(\!|}_{x} \mathcal{B} \mathop{|\!)}^{y} \ , \tag{5.2}$$

$$\mathsf{mb\_vpre}_{\theta_\mathcal{B}}(p) \leqslant \mathop{(\!|}_{x} \mathcal{B} \mathop{|\!)}^{u} \ , \tag{5.3}$$

$$\mathsf{mb\_vpre}_{\theta_\mathcal{B}}(p) \leqslant \mathop{(\!|}_{u} \mathcal{B} \mathop{|\!)}^{u} \ , \tag{5.4}$$

$$\mathsf{mb\_vpre}_{\theta_\mathcal{B}}(p) \leqslant \mathop{(\!|}_{u} \mathcal{B} \mathop{|\!)}^{y} \ . \tag{5.5}$$

By the definition of $\mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}$, we have to prove that

$$(\mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(p))^* \leqslant \underset{x}{\overset{y}{(\!|\,\mathcal{B}\,|\!)}} \ .$$

By Kleene algebra, it suffices to prove that

$$1 \leqslant \underset{x}{\overset{y}{(\!|\,\mathcal{B}\,|\!)}} \ , \tag{5.6}$$

$$\mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(p) \leqslant \underset{x}{\overset{y}{(\!|\,\mathcal{B}\,|\!)}} \ , \tag{5.7}$$

$$\mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(p) \cdot \mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(p) \leqslant \underset{x}{\overset{y}{(\!|\,\mathcal{B}\,|\!)}} \ , \tag{5.8}$$

$$\mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(p) \cdot \mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(p) \cdot (\mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(p))^* \cdot \mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(p) \leqslant \underset{x}{\overset{y}{(\!|\,\mathcal{B}\,|\!)}} \ . \tag{5.9}$$

Inequation (5.6) is trivial by axiom (3.2) and the fact that $\lfloor_x 1 \rfloor^y$ is "hidden" in $\lfloor_x p^* \rfloor^y$ that is in $\mathcal{B}$.

Inequation (5.7) is simply the inductive hypothesis (5.2).

Inequation (5.8) is proved by using inductive hypotheses (5.3) and (5.5), Kleene algebra and axiom (3.4) extended to metablocks.

For (5.9), by axiom (3.4) extended to metablocks, by the inductive hypotheses (5.3) and (5.5), and by Kleene algebra, it suffices to prove that

$$\begin{aligned}\mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(p) \cdot \mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(p) \cdot (\mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(p))^* \cdot \mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(p) \\ \leqslant \mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(p) \cdot \underset{u}{\overset{u}{(\!|\,\mathcal{B}\,|\!)}} \cdot \mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(p) \ .\end{aligned}$$

By monotonicity of $\cdot$, it suffices to prove that

$$\mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(p) \cdot (\mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(p))^* \leqslant \underset{u}{\overset{u}{(\!|\,\mathcal{B}\,|\!)}} \ .$$

Thus, by the Kleene star induction axiom and by simple Kleene algebraic reasoning, it suffices to prove that

$$\begin{aligned}\mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(p) &\leqslant \underset{u}{\overset{u}{(\!|\,\mathcal{B}\,|\!)}} \ , \\ \underset{u}{\overset{u}{(\!|\,\mathcal{B}\,|\!)}} \cdot \mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(p) &\leqslant \underset{u}{\overset{u}{(\!|\,\mathcal{B}\,|\!)}} \ .\end{aligned}$$

The first inequation is direct from inductive hypothesis (5.4). The second inequation is proved by using inductive hypothesis (5.4), Kleene algebra and axiom (3.4) extended to metablocks. ■

We now want to extend the induction axioms (3.14) and (3.15) to metablocks. We first define some functions. Let $\mathsf{set\_labels} : \mathsf{MBexp} \to 2^{V \times V}$ be a function that extracts all explicit pairs of labels in a metablock expression. It is defined inductively by:

- $\mathsf{set\_labels}(a) := \emptyset$ for $a \in \Sigma_{\mathbf{i}} \cup \mathsf{Tests}_{\mathbf{B}}$;
- $\mathsf{set\_labels}((c{\downarrow_z}{\uparrow^w} r)) := \{(z, w)\}$ for $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $z, w \in V$;
- $\mathsf{set\_labels}(p \cdot q) := \mathsf{set\_labels}(p) \cup \mathsf{set\_labels}(q)$;
- $\mathsf{set\_labels}(p + q) := \mathsf{set\_labels}(p) \cup \mathsf{set\_labels}(q)$;
- $\mathsf{set\_labels}(p^*) := \mathsf{set\_labels}(p)$.

Also, let $\mathsf{FM}^*_{\mathcal{B}}$ and $\mathsf{BM}^*_{\mathcal{B}}$ be functions that approximate the needed metablocks for the induction axioms. These functions are the least fixed points of the monotone functions $\mathsf{FM}^1_{\mathcal{B}} : 2^{V \times V} \to 2^{V \times V}$ and $\mathsf{BM}^1_{\mathcal{B}} : 2^{V \times V} \to 2^{V \times V}$ defined for every $T \subseteq V \times V$ by:

$$\mathsf{FM}^1_{\mathcal{B}}(T) := T \cup \{(y, y'), (w, w') \mid (\exists\ z, m \mid (z, y') \in T : {}_z[\, m \,]^y \in \mathcal{B}^{\mathsf{set}} \\ \wedge\ (w, w') \in \mathsf{set\_labels}(m))\}\ ,$$

$$\mathsf{BM}^1_{\mathcal{B}}(T) := T \cup \{(y, y'), (w, w') \mid (\exists\ z, m \mid (y, z) \in T : {}_{y'}[\, m \,]^z \in \mathcal{B}^{\mathsf{set}} \\ \wedge\ (w, w') \in \mathsf{set\_labels}(m))\}\ .$$

Now, let $\vartheta_{\mathcal{B}}$ be any substitution environment that is defined for all pairs $(u, u') \in \mathsf{FM}^*_{\mathcal{B}}(\{(x, y)\})$ by a visibly pushdown regular expression. Then, a generalization of the induction axiom (3.14) can be proved:

$$\boxed{\begin{array}{l} \Big( \wedge\, u, u' \mid (u, u') \in \mathsf{FM}^*_{\mathcal{B}}(\{(x, y)\}) : \\ \qquad (\wedge\, m \mid {}^{u'}_{u}[\, m \,] \in \mathcal{B}^{\mathsf{set}} : \mathsf{mb\_vpre}_{\vartheta_{\mathcal{B}}}(m) \leqslant \vartheta_{\mathcal{B}}(u, u')) \\ \quad \wedge\ (\wedge\, m, v \mid {}^{v}_{u}[\, m \,] \in \mathcal{B}^{\mathsf{set}} : \mathsf{mb\_vpre}_{\vartheta_{\mathcal{B}}}(m) \cdot \vartheta_{\mathcal{B}}(v, u') \leqslant \vartheta_{\mathcal{B}}(u, u')) \Big) \\ \to\ {}^{y}_{x}(\!|\, \mathcal{B} \,|\!) \leqslant \vartheta_{\mathcal{B}}(x, y)\ . \end{array}} \tag{5.10}$$

Now, let $\vartheta'_{\mathcal{B}}$ be any substitution environment that is defined for all pairs $(u, u') \in \mathsf{BM}^*_{\mathcal{B}}(\{(x, y)\})$ by a visibly pushdown regular expression. Then, a generalization of the

induction axiom (3.15) can be proved:

$$\boxed{\begin{array}{l}\Big(\wedge\, u, u' \mid (u,u') \in \mathsf{BM}^*_{\mathcal{B}}(\{(x,y)\}) : \\ \qquad (\wedge\, m \mid \underset{u}{[}\, m \,\overset{u'}{]} \in \mathcal{B}^{\mathsf{set}} : \mathsf{mb\_vpre}_{\vartheta'_{\mathcal{B}}}(m) \leqslant \vartheta'_{\mathcal{B}}(u,u')) \\ \quad \wedge\ (\wedge\, m, v \mid \underset{v}{[}\, m \,\overset{u'}{]} \in \mathcal{B}^{\mathsf{set}} : \vartheta'_{\mathcal{B}}(u,v) \cdot \mathsf{mb\_vpre}_{\vartheta'_{\mathcal{B}}}(m) \leqslant \vartheta'_{\mathcal{B}}(u,u'))\Big) \\ \rightarrow\ \underset{x}{(\!|}\, \mathcal{B} \,\overset{y}{|\!)} \leqslant \vartheta'_{\mathcal{B}}(x,y)\ .\end{array}} \tag{5.11}$$

We just prove (5.10) since the proof of (5.11) is similar. We first define a function that will be very useful. Let $\mathsf{mb\_vpre\_suffixes}_\theta$ be a function that takes a metablock and generates a partial function. The partial function takes a starting label of a block contained in the metablock under $\mathsf{mb}$, and returns a visibly pushdown regular expression representing the suffix of the metablock from the label. The function $\mathsf{mb\_vpre\_suffixes}_\theta$ is defined inductively by:

- if $a \in \Sigma_{\mathbf{i}} \cup \mathsf{Tests}_{\mathbf{B}}$, and $x$ and $y$ are labels, then
$$\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, a \,\overset{y}{]}) := \{x \mapsto a\}\ ;$$
- if $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$, and $x$, $y$, $z$ and $w$ are labels, then
$$\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}(c \underset{z}{\downarrow} \overset{w}{\uparrow} r) \,\overset{y}{]}) := \{x \mapsto c \cdot \theta(z,w) \cdot r\}\ ;$$
- if $x$ and $y$ are labels and $z$ is a fresh label, then
$$\begin{array}{l}\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \cdot q \,\overset{y}{]}) := \\ \qquad \{w \mapsto p' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{z}{[}\, q \,\overset{y}{]}))(z) \mid (w \mapsto p') \in \mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \,\overset{z}{]})\} \\ \qquad \cup\, \mathsf{mb\_vpre\_suffixes}_\theta(\underset{z}{[}\, q \,\overset{y}{]})\ ;\end{array}$$
- if $x$ and $y$ are labels, then
$$\begin{array}{l}\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p + q \,\overset{y}{]}) := \\ \qquad \{x \mapsto (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \,\overset{y}{]}))(x) + (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, q \,\overset{y}{]}))(x)\} \\ \qquad \cup \{w \mapsto p' \mid (w \mapsto p') \in \mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \,\overset{y}{]}) \wedge w \neq x\} \\ \qquad \cup \{w \mapsto q' \mid (w \mapsto q') \in \mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, q \,\overset{y}{]}) \wedge w \neq x\}\ ;\end{array}$$

- if $x$ and $y$ are labels and $z$ is a fresh label, then

$$\begin{aligned}
&\mathsf{mb_vpre_suffixes}_\theta(\overset{y}{\underset{x}{[}}\, p^* \,\overset{y}{\underset{x}{]}}) := \\
&\quad \{x \mapsto ((\mathsf{mb_vpre_suffixes}_\theta(\overset{y}{\underset{x}{[}}\, p \,\overset{y}{\underset{x}{]}}))(x))^*\} \\
&\quad \cup \{z \mapsto ((\mathsf{mb_vpre_suffixes}_\theta(\overset{z}{\underset{z}{[}}\, p \,\overset{z}{\underset{z}{]}}))(z))^* \cdot (\mathsf{mb_vpre_suffixes}_\theta(\overset{y}{\underset{z}{[}}\, p \,\overset{y}{\underset{z}{]}}))(z)\} \\
&\quad \cup \{w \mapsto p' \mid (w \mapsto p') \in \mathsf{mb_vpre_suffixes}_\theta(\overset{y}{\underset{x}{[}}\, p \,\overset{y}{\underset{x}{]}}) \wedge w \neq x\} \\
&\quad \cup \{w \mapsto p' \cdot ((\mathsf{mb_vpre_suffixes}_\theta(\overset{z}{\underset{z}{[}}\, p \,\overset{z}{\underset{z}{]}}))(z))^* \cdot (\mathsf{mb_vpre_suffixes}_\theta(\overset{y}{\underset{z}{[}}\, p \,\overset{y}{\underset{z}{]}}))(z) \\
&\qquad \mid (w \mapsto p') \in \mathsf{mb_vpre_suffixes}_\theta(\overset{z}{\underset{x}{[}}\, p \,\overset{z}{\underset{x}{]}}) \wedge w \neq x\} \\
&\quad \cup \{w \mapsto p' \cdot ((\mathsf{mb_vpre_suffixes}_\theta(\overset{z}{\underset{z}{[}}\, p \,\overset{z}{\underset{z}{]}}))(z))^* \cdot (\mathsf{mb_vpre_suffixes}_\theta(\overset{y}{\underset{z}{[}}\, p \,\overset{y}{\underset{z}{]}}))(z) \\
&\qquad \mid (w \mapsto p') \in \mathsf{mb_vpre_suffixes}_\theta(\overset{z}{\underset{z}{[}}\, p \,\overset{z}{\underset{z}{]}}) \wedge w \neq z\} \\
&\quad \cup \{w \mapsto p' \mid (w \mapsto p') \in \mathsf{mb_vpre_suffixes}_\theta(\overset{y}{\underset{z}{[}}\, p \,\overset{y}{\underset{z}{]}}) \wedge w \neq z\} \ .
\end{aligned}$$

We now present three results on the function $\mathsf{mb\_vpre\_suffixes}_\theta$. The proof of these three results (lemmas) is given in Appendix F.

**Lemma 5.5** (Relation between $\mathsf{mb\_vpre\_suffixes}_\theta$ and $\mathsf{mb\_vpre}_\theta$)**.** *For all metablocks* $\overset{y}{\underset{x}{[}}\, m \,\overset{y}{\underset{x}{]}}$,

$$(\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}}\, m \,\overset{y}{\underset{x}{]}}))(x) = \mathsf{mb\_vpre}_\theta(m) \ . \tag{5.12}$$

**Lemma 5.6** (Handling unary and binary blocks that *end with* the same label as their associated metablock)**.** *For all metablocks* $\overset{y}{\underset{x}{[}}\, m \,\overset{y}{\underset{x}{]}}$, *unary blocks* $[_z\, m' \,]^y \in \mathsf{mb}(\overset{y}{\underset{x}{[}}\, m \,\overset{y}{\underset{x}{]}})^1$ *and binary blocks* $[_{z'}\, c{\downarrow_w}{\uparrow^{w'}}\, r \,]^y \in \mathsf{mb}(\overset{y}{\underset{x}{[}}\, m \,\overset{y}{\underset{x}{]}})^2$,

$$m' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}}\, m \,\overset{y}{\underset{x}{]}}))(z) \ , \tag{5.13}$$

$$c \cdot \theta(w, w') \cdot r \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}}\, m \,\overset{y}{\underset{x}{]}}))(z') \ . \tag{5.14}$$

**Lemma 5.7** (Handling unary and binary blocks that *do not end with* the same label as their associated metablock.)**.** *For all metablocks* $\overset{y}{\underset{x}{[}}\, m \,\overset{y}{\underset{x}{]}}$, *unary blocks* $[_z\, m' \,]^{z'} \in$

$\mathsf{mb}(\llbracket_x\, m\,\rrbracket^y)^1$ *and binary blocks* $\llbracket_u\, c \downarrow_w \uparrow^{w'} r\,\rrbracket^{u'} \in \mathsf{mb}(\llbracket_x\, m\,\rrbracket^y)^2$ *such that* $z' \neq y$ *and* $u' \neq y$,

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{\llbracket}}\, m\,\rrbracket))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{\llbracket}}\, m\,\rrbracket))(z) \ , \tag{5.15}$$

$$\begin{aligned} c \cdot \theta(w, w') \cdot r \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{\llbracket}}\, m\,\rrbracket))(u') \\ \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{\llbracket}}\, m\,\rrbracket))(u) \ . \end{aligned} \tag{5.16}$$

We are now ready to prove (5.10).

*Proof.* We suppose that, for all $(u, u') \in \mathsf{FM}^*_{\mathcal{B}}(\{(x, y)\})$,

- each metablock $\llbracket_u\, m\,\rrbracket^{u'} \in \mathcal{B}^{\mathsf{set}}$ is such that

$$\mathsf{mb\_vpre}_{\vartheta_{\mathcal{B}}}(m) \leqslant \vartheta_{\mathcal{B}}(u, u') \ , \tag{5.17}$$

- each metablock $\llbracket_u\, m\,\rrbracket^v \in \mathcal{B}^{\mathsf{set}}$ (for every $v \in V$) is such that

$$\mathsf{mb\_vpre}_{\vartheta_{\mathcal{B}}}(m) \cdot \vartheta_{\mathcal{B}}(v, u') \leqslant \vartheta_{\mathcal{B}}(u, u') \ , \tag{5.18}$$

and we must prove that

$$\overset{y}{\underset{x}{(\!|\, \mathcal{B}\,|\!)}} \leqslant \vartheta_{\mathcal{B}}(x, y) \ . \tag{5.19}$$

Note that $\mathsf{FM}^*_{\mathcal{B}}(\{(x, y)\}) \subseteq \mathsf{F}^*_{\mathsf{mb}(\mathcal{B})}(\{(x, y)\})$, but it is not necessarily the case that $\mathsf{F}^*_{\mathsf{mb}(\mathcal{B})}(\{(x, y)\}) \subseteq \mathsf{FM}^*_{\mathcal{B}}(\{(x, y)\})$. So, we cannot use directly axiom (3.14) since $\vartheta_{\mathcal{B}}$ may be undefined for some pairs of $\mathsf{F}^*_{\mathsf{mb}(\mathcal{B})}(\{(x, y)\})$.

To facilitate the discussion, let $V'$ be the set of fresh labels used in the generation of $\mathsf{mb}(\mathcal{B})$. Note that, by definition of $\mathsf{mb}$, each label $z \in V'$ is used in the transformation of *one and only one* metablock of $\mathcal{B}$ into a list of blocks. Note also that the pairs in $\mathsf{F}^*_{\mathsf{mb}(\mathcal{B})}(\{(x, y)\}) \backslash \mathsf{FM}^*_{\mathcal{B}}(\{(x, y)\})$ are all of the form $(z, z')$ where $z \in V'$ and $z' \in V$.

Let us define a "saturation" of the substitution environment $\vartheta_{\mathcal{B}}$ for the pairs of $\mathsf{F}^*_{\mathsf{mb}(\mathcal{B})}(\{(x, y)\})$. We first define ${\vartheta_{\mathcal{B}}}^{\mathsf{sat}}$ by

$${\vartheta_{\mathcal{B}}}^{\mathsf{sat}}(u, u') := \vartheta_{\mathcal{B}}(u, u')$$

for all $(u, u') \in \mathsf{FM}^*_{\mathcal{B}}(\{(x, y)\})$.

For the remaining pairs $(u,u') \in \mathsf{F}^*_{\mathsf{mb}(\mathcal{B})}(\{(x,y)\}) \backslash \mathsf{FM}^*_{\mathcal{B}}(\{(x,y)\})$, let $\lfloor_z\, m' \,\rfloor^{z'} \in \mathcal{B}^{\mathsf{set}}$ be the unique metablock that produced the fresh label $u$. We define ${\vartheta_{\mathcal{B}}}^{\mathsf{sat}}(u,u')$ by

$${\vartheta_{\mathcal{B}}}^{\mathsf{sat}}(u,u') := \begin{cases} (\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\lfloor_z\, m' \,\rfloor^{z'}))(u) \cdot (1 + \vartheta_{\mathcal{B}}(u',u')) & \text{if } z' = u', \\ (\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\lfloor_z\, m' \,\rfloor^{z'}))(u) \cdot \vartheta_{\mathcal{B}}(z',u') & \text{otherwise.} \end{cases}$$

To prove (5.19), by axiom (3.14) with solutions ${\vartheta_{\mathcal{B}}}^{\mathsf{sat}}$, it suffices to prove that, for all $(u,u') \in \mathsf{F}^*_{\mathsf{mb}(\mathcal{B})}(\{(x,y)\})$,

- for each unary block $\lfloor_u\, m \,\rfloor^{u'} \in \mathsf{mb}(\mathcal{B})^1$,

$$m \leqslant {\vartheta_{\mathcal{B}}}^{\mathsf{sat}}(u,u') \ , \tag{5.20}$$

- for each unary block $\lfloor_u\, m \,\rfloor^{v} \in \mathsf{mb}(\mathcal{B})^1$ where $v \in V \cup V'$,

$$m \cdot {\vartheta_{\mathcal{B}}}^{\mathsf{sat}}(v,u') \leqslant {\vartheta_{\mathcal{B}}}^{\mathsf{sat}}(u,u') \ , \tag{5.21}$$

- for each binary block $\lfloor_u\, c{\downarrow_z}{\uparrow^w}\, r \,\rfloor^{u'} \in \mathsf{mb}(\mathcal{B})^2$,

$$c \cdot \vartheta_{\mathcal{B}}(z,w) \cdot r \leqslant {\vartheta_{\mathcal{B}}}^{\mathsf{sat}}(u,u') \ , \tag{5.22}$$

- for each binary block $\lfloor_u\, c{\downarrow_z}{\uparrow^w}\, r \,\rfloor^{v} \in \mathsf{mb}(\mathcal{B})^2$ where $v \in V \cup V'$,

$$c \cdot \vartheta_{\mathcal{B}}(z,w) \cdot r \cdot {\vartheta_{\mathcal{B}}}^{\mathsf{sat}}(v,u') \leqslant {\vartheta_{\mathcal{B}}}^{\mathsf{sat}}(u,u') \ . \tag{5.23}$$

Note that, in (5.22) and (5.23), we use $\vartheta_{\mathcal{B}}(z,w)$ instead of ${\vartheta_{\mathcal{B}}}^{\mathsf{sat}}(z,w)$. The reason is that $z$ and $w$ are in $V$ by the definition of the functions $\mathsf{FM}^*_{\mathcal{B}}$ and $\mathsf{F}^*_{\mathsf{mb}(\mathcal{B})}$. So, for this case,

$${\vartheta_{\mathcal{B}}}^{\mathsf{sat}}(z,w) = \vartheta_{\mathcal{B}}(z,w) \ .$$

For (5.20), let $\lfloor_t\, m' \,\rfloor^{u'} \in \mathcal{B}^{\mathsf{set}}$ be the metablock that produced $\lfloor_u\, m \,\rfloor^{u'}$. By Lemma 5.6, inequation (5.13), we have that

$$m \leqslant (\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\underset{t}{\lfloor}\, m' \,\rfloor^{u'}))(u) \ .$$

So, to prove (5.20), by Kleene algebra, it suffices to show that

$$(\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\underset{t}{\lfloor}\, m' \,\rfloor^{u'}))(u) \leqslant {\vartheta_{\mathcal{B}}}^{\mathsf{sat}}(u,u') \ .$$

Let us do a proof by case analysis on $u$. If $u \in V$, then we must prove that

$$(\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\underset{t}{[}\, m' \,\overset{u'}{]}))(u) \leqslant \vartheta_{\mathcal{B}}(u, u') \ .$$

By the definition of $\mathsf{mb}$, this case happens only if $t = u$. So, by Lemma (5.5), it suffices to prove that

$$\mathsf{mb\_vpre}_{\vartheta_{\mathcal{B}}}(m') \leqslant \vartheta_{\mathcal{B}}(t, u') \ .$$

This is direct from hypothesis (5.17).

If $u \in V'$, then we must prove that

$$\begin{aligned}&(\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\underset{t}{[}\, m' \,\overset{u'}{]}))(u)\\ &\leqslant (\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\underset{t}{[}\, m' \,\overset{u'}{]}))(u) \cdot (1 + \vartheta_{\mathcal{B}}(u', u')) \ .\end{aligned}$$

This is trivial by Kleene algebra.

For (5.21), let $\underset{t}{[}\, m' \,\overset{t'}{]} \in \mathcal{B}^{\mathsf{set}}$ be the metablock that produced $\underset{u}{[}\, m \,\overset{v}{]}$. Let us do a proof by case analysis. If $u \in V$ and $v \in V$, then we must prove that

$$m \cdot \vartheta_{\mathcal{B}}(v, u') \leqslant \vartheta_{\mathcal{B}}(u, u') \ .$$

By the definition of $\mathsf{mb}$, this case happens only if $t = u$ and $t' = v$. By Lemma 5.6, inequation (5.13), and by Kleene algebra, it suffices to prove that

$$(\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\underset{u}{[}\, m' \,\overset{v}{]}))(u) \cdot \vartheta_{\mathcal{B}}(v, u') \leqslant \vartheta_{\mathcal{B}}(u, u') \ .$$

So, by Lemma 5.5, it suffices to prove that

$$\mathsf{mb\_vpre}_{\vartheta_{\mathcal{B}}}(m') \cdot \vartheta_{\mathcal{B}}(v, u') \leqslant \vartheta_{\mathcal{B}}(u, u') \ .$$

This is direct from hypothesis (5.18).

If $u \in V'$, $v \in V$ and $v = u'$, then, by the definition of $\mathsf{mb}$, this case happens only if $t' = v$, and we must prove that

$$m \cdot \vartheta_{\mathcal{B}}(u', u') \leqslant (\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\underset{t}{[}\, m' \,\overset{u'}{]}))(u) \cdot (1 + \vartheta_{\mathcal{B}}(u', u')) \ .$$

By Lemma 5.6, inequation (5.13), and by Kleene algebra, it suffices to prove that

$$\begin{aligned}&(\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\underset{t}{[}\, m' \,\overset{u'}{]}))(u) \cdot \vartheta_{\mathcal{B}}(u', u')\\ &\leqslant (\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\underset{t}{[}\, m' \,\overset{u'}{]}))(u) \cdot (1 + \vartheta_{\mathcal{B}}(u', u')) \ .\end{aligned}$$

This is trivial by Kleene algebra.

If $u \in V'$, $v \in V$ and $v \neq u'$, then, by the definition of $\mathsf{mb}$, this case happens only if $t' = v$, and we must prove that

$$m \cdot \vartheta_{\mathcal{B}}(v, u') \leqslant (\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}({\underset{t}{[}} m' {\overset{v}{]}}))(u) \cdot \vartheta_{\mathcal{B}}(v, u') \ .$$

By monotonicity of $\cdot$, it suffices to prove that

$$m \leqslant (\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}({\underset{t}{[}} m' {\overset{v}{]}}))(u) \ .$$

This is direct by Lemma 5.6, inequation (5.13).

If $u \in V$, $v \in V'$ and $t' = u'$, then, by the definition of $\mathsf{mb}$, this case happens only if $t = u$, and we must prove that

$$m \cdot (\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}({\underset{u}{[}} m' {\overset{u'}{]}}))(v) \cdot (1 + \vartheta_{\mathcal{B}}(u', u')) \leqslant \vartheta_{\mathcal{B}}(u, u') \ .$$

By Lemma 5.7, inequation (5.15), and by Kleene algebra, it suffices to prove that

$$(\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}({\underset{u}{[}} m' {\overset{u'}{]}}))(u) \cdot (1 + \vartheta_{\mathcal{B}}(u', u')) \leqslant \vartheta_{\mathcal{B}}(u, u') \ .$$

So, by Lemma 5.5, it suffices to prove that

$$\mathsf{mb\_vpre}_{\vartheta_{\mathcal{B}}}(m') \cdot (1 + \vartheta_{\mathcal{B}}(u', u')) \leqslant \vartheta_{\mathcal{B}}(u, u') \ .$$

By Kleene algebra, it suffices to prove the two following formulae:

$$\begin{aligned} &\mathsf{mb\_vpre}_{\vartheta_{\mathcal{B}}}(m') \leqslant \vartheta_{\mathcal{B}}(u, u') \ , \\ &\mathsf{mb\_vpre}_{\vartheta_{\mathcal{B}}}(m') \cdot \vartheta_{\mathcal{B}}(u', u') \leqslant \vartheta_{\mathcal{B}}(u, u') \ . \end{aligned}$$

This is direct from hypotheses (5.17) and (5.18).

If $u \in V$, $v \in V'$ and $t' \neq u'$, then, by the definition of $\mathsf{mb}$, this case happens only if $t = u$, and we must prove that

$$m \cdot (\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}({\underset{u}{[}} m' {\overset{t'}{]}}))(v) \cdot \vartheta_{\mathcal{B}}(t', u') \leqslant \vartheta_{\mathcal{B}}(u, u') \ .$$

By Lemma 5.7, inequation (5.15), and by Kleene algebra, it suffices to prove that

$$(\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}({\underset{u}{[}} m' {\overset{t'}{]}}))(u) \cdot \vartheta_{\mathcal{B}}(t', u') \leqslant \vartheta_{\mathcal{B}}(u, u') \ .$$

So, by Lemma 5.5, it suffices to prove that

$$\mathsf{mb\_vpre}_{\vartheta_{\mathcal{B}}}(m') \cdot \vartheta_{\mathcal{B}}(t', u') \leqslant \vartheta_{\mathcal{B}}(u, u') \ .$$

This is direct from hypothesis (5.18).

If $u \in V'$, $v \in V'$ and $t' = u'$, then we must prove that

$$\begin{aligned} & m \cdot (\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\underset{t}{[}\, m' \,\overset{u'}{]}))(v) \cdot (1 + \vartheta_{\mathcal{B}}(u', u')) \\ & \leqslant (\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\underset{t}{[}\, m' \,\overset{u'}{]}))(u) \cdot (1 + \vartheta_{\mathcal{B}}(u', u')) \ . \end{aligned}$$

By monotonicity of $\cdot$, it suffices to prove that

$$m \cdot (\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\underset{t}{[}\, m' \,\overset{u'}{]}))(v) \leqslant (\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\underset{t}{[}\, m' \,\overset{u'}{]}))(u) \ .$$

This is direct from Lemma 5.7, inequation (5.15).

If $u \in V'$, $v \in V'$ and $t' \neq u'$, then we must prove that

$$\begin{aligned} & m \cdot (\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\underset{t}{[}\, m' \,\overset{t'}{]}))(v) \cdot \vartheta_{\mathcal{B}}(t', u') \\ & \leqslant (\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\underset{t}{[}\, m' \,\overset{t'}{]}))(u) \cdot \vartheta_{\mathcal{B}}(t', u') \ . \end{aligned}$$

By monotonicity of $\cdot$, it suffices to prove that

$$m \cdot (\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\underset{t}{[}\, m' \,\overset{t'}{]}))(v) \leqslant (\mathsf{mb\_vpre\_suffixes}_{\vartheta_{\mathcal{B}}}(\underset{t}{[}\, m' \,\overset{t'}{]}))(u) \ .$$

This is direct from Lemma 5.7, inequation (5.15).

The proof of (5.22) is similar to the proof of (5.20).

The proof of (5.23) is similar to the proof of (5.21). ■

We now present some laws that can be derived from the extension of the axioms to metablocks. We first show an extension of (3.22) to metablocks. Let $\theta_{\mathcal{B}}$ be the substitution environment defined, for all $u, u' \in V$, by

$$\theta_{\mathcal{B}}(u, u') := \underset{u}{(\!|}\, \mathcal{B} \,\overset{u'}{|\!)} \ .$$

Then it can be proved that

$$\begin{aligned}\mathop{(\!|}_{x}^{y}\mathcal{B}\mathop{|\!)}_{x}^{y} = (\sum\, m \mid \mathop{[\,m\,]}_{x}^{y} \in \mathcal{B}^{\mathsf{set}} : \mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(m)) \\ +\ (\sum\, m, v \mid \mathop{[\,m\,]}_{x}^{v} \in \mathcal{B}^{\mathsf{set}} : \mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(m) \cdot \mathop{(\!|}_{v}^{y}\mathcal{B}\mathop{|\!)}_{v}^{y})\ .\end{aligned} \tag{5.24}$$

*Proof.* The case $\geqslant$ is direct from Kleene algebra, inequation (5.1) and axiom (3.4) extended to metablocks.

For the case $\leqslant$, we use (5.10) with solutions from $\theta_{\mathcal{B}}$. So, it suffices to prove that, for all $u, u' \in V$,

- for every metablock $\mathop{[\,m\,]}_{u}^{u'} \in \mathcal{B}^{\mathsf{set}}$,

$$\mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(m) \leqslant \theta_{\mathcal{B}}(u, u')\ ;$$

- for every metablock $\mathop{[\,m\,]}_{u}^{v} \in \mathcal{B}^{\mathsf{set}}$ where $v \in V$,

$$\mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(m) \cdot \theta_{\mathcal{B}}(v, u') \leqslant \theta_{\mathcal{B}}(u, u')\ .$$

By the definition of $\theta_{\mathcal{B}}$, it suffices to prove that, for all $u, u' \in V$,

- for every metablock $\mathop{[\,m\,]}_{u}^{u'} \in \mathcal{B}^{\mathsf{set}}$,

$$\mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(m) \leqslant \mathop{(\!|}_{u}^{u'}\mathcal{B}\mathop{|\!)}_{u}^{u'}\ ;$$

- for every metablock $\mathop{[\,m\,]}_{u}^{v} \in \mathcal{B}^{\mathsf{set}}$ where $v \in V$,

$$\mathsf{mb\_vpre}_{\theta_{\mathcal{B}}}(m) \cdot \mathop{(\!|}_{v}^{u'}\mathcal{B}\mathop{|\!)}_{v}^{u'} \leqslant \mathop{(\!|}_{u}^{u'}\mathcal{B}\mathop{|\!)}_{u}^{u'}\ .$$

This is direct from inequation (5.1), Kleene algebra and axiom (3.4) extended to metablocks. ■

Another useful law allows one to replace the "content" of a metablock by another content if their associated visibly pushdown regular expressions are equal. Let $\mathcal{C}$ be the same list of metablocks as $\mathcal{B}$ except for a metablock $\mathop{[\,m_1\,]}_{z}^{w}$ in $\mathcal{B}^{\mathsf{set}}$ that becomes

a metablock $\llbracket_z\, m_2 \rrbracket^w$ in $\mathcal{C}^{\mathsf{set}}$. Let $\theta'$ be the substitution environment defined by, for all $u, u' \in V$,

$$\theta'(u, u') := \underset{u}{\overset{u'}{(\!|\, \mathcal{C} \,|\!)}} \quad .$$

Then, it can be proved that

$$\mathsf{mb\_vpre}_{\theta'}(m_1) = \mathsf{mb\_vpre}_{\theta'}(m_2) \to \underset{x}{\overset{y}{(\!|\, \mathcal{B} \,|\!)}} = \underset{x}{\overset{y}{(\!|\, \mathcal{C} \,|\!)}} \quad . \tag{5.25}$$

Note that $\mathsf{mb\_vpre}_{\theta'}(m_1)$ is well defined since $\mathcal{B}$ and $\mathcal{C}$ share the same labels.

Of course, we also have that

$$\mathsf{mb\_vpre}_{\theta'}(m_1) \leqslant \mathsf{mb\_vpre}_{\theta'}(m_2) \to \underset{x}{\overset{y}{(\!|\, \mathcal{B} \,|\!)}} \leqslant \underset{x}{\overset{y}{(\!|\, \mathcal{C} \,|\!)}} \quad . \tag{5.26}$$

In fact, inequation (5.26) is sufficient to prove (5.25) since the other case is similar.

*Proof.* We suppose

$$\mathsf{mb\_vpre}_{\theta'}(m_1) \leqslant \mathsf{mb\_vpre}_{\theta'}(m_2) \tag{5.27}$$

and we prove

$$\underset{x}{\overset{y}{(\!|\, \mathcal{B} \,|\!)}} \leqslant \underset{x}{\overset{y}{(\!|\, \mathcal{C} \,|\!)}} \quad . \tag{5.28}$$

To prove (5.28), we use (5.10) with solutions from $\theta'$. So, it suffices to prove that, for all $u, u' \in V$,

- for every metablock $\llbracket_u\, m \rrbracket^{u'} \in \mathcal{B}^{\mathsf{set}}$,

$$\mathsf{mb\_vpre}_{\theta'}(m) \leqslant \theta'(u, u') \ ;$$

- for every metablock $\llbracket_u\, m \rrbracket^{v} \in \mathcal{B}^{\mathsf{set}}$ where $v \in V$,

$$\mathsf{mb\_vpre}_{\theta'}(m) \cdot \theta'(v, u') \leqslant \theta'(u, u') \quad .$$

By the definition of $\theta'$, it suffices to prove that, for all $u, u' \in V$,

- for every metablock $\llbracket_u\, m \rrbracket^{u'} \in \mathcal{B}^{\mathsf{set}}$,

$$\mathsf{mb\_vpre}_{\theta'}(m) \leqslant \underset{u}{\overset{u'}{(\!|\, \mathcal{C} \,|\!)}} \ ; \tag{5.29}$$

- for every metablock $\left[_u\, m\right]^v \in \mathcal{B}^{\mathsf{set}}$ where $v \in V$,

$$\mathsf{mb\_vpre}_{\theta'}(m) \cdot \underset{v}{(\!|}\, \mathcal{C}\, \overset{u'}{|\!)} \leqslant \underset{u}{(\!|}\, \mathcal{C}\, \overset{u'}{|\!)} \quad . \tag{5.30}$$

For all metablocks $\left[_u\, m\right]^v \in \mathcal{B}^{\mathsf{set}}$ (except for $\left[_z\, m_1\right]^w$), the same metablock exists in $\mathcal{C}^{\mathsf{set}}$. So, inequations (5.29) and (5.30) are trivial for these cases by (5.1), Kleene algebra and axiom (3.4) extended to metablocks. It remains to prove that for the metablock $\left[_z\, m_1\right]^w$,

$$\mathsf{mb\_vpre}_{\theta'}(m_1) \leqslant \underset{z}{(\!|}\, \mathcal{C}\, \overset{w}{|\!)} \ ;$$

$$\mathsf{mb\_vpre}_{\theta'}(m_1) \cdot \underset{w}{(\!|}\, \mathcal{C}\, \overset{u'}{|\!)} \leqslant \underset{z}{(\!|}\, \mathcal{C}\, \overset{u'}{|\!)} \quad .$$

By definition of $\mathcal{C}$ and by hypothesis (5.27), we have that there exists a metablock $\left[_z\, m_2\right]^w$ in $\mathcal{C}^{\mathsf{set}}$ such that

$$\mathsf{mb\_vpre}_{\theta'}(m_1) \leqslant \mathsf{mb\_vpre}_{\theta'}(m_2) \quad .$$

So, by Kleene algebra, it suffices to prove that

$$\mathsf{mb\_vpre}_{\theta'}(m_2) \leqslant \underset{z}{(\!|}\, \mathcal{C}\, \overset{w}{|\!)} \ ;$$

$$\mathsf{mb\_vpre}_{\theta'}(m_2) \cdot \underset{w}{(\!|}\, \mathcal{C}\, \overset{u'}{|\!)} \leqslant \underset{z}{(\!|}\, \mathcal{C}\, \overset{u'}{|\!)} \quad .$$

This is trivial by (5.1), Kleene algebra and axiom (3.4) extended to metablocks. ■

## 5.4 New Axioms for Visibly Pushdown Kleene Algebra with Tests: Propagation of Tests

When working with programs and tests, it is sometimes useful to guard a term by tests. For example, knowing whether a test $a$ is true before doing an internal action $p$ (in other words, $a$ is a precondition of $p$) can result in knowing whether a test $b$ is true after doing $p$ (in other words, $b$ is the postcondition of $p$ when respecting $a$). This is usually expressed in Kleene algebra with tests by an hypothesis $ap \leqslant pb$ (or, equivalently, $ap = apb$) representing the Hoare triple $\{a\}\ p\ \{b\}$. These hypotheses are an important tool when dealing with program analysis. During a proof attempt, most of the time, we end up handling terms like $apb$ (we call them *guarded terms*).

However, it seems hard to guard operands in unary and binary blocks with tests. In fact, it is easy to do when dealing with "non-recursive" blocks, but it is difficult to deal with recursive ones[2]. To ease this process, we introduce axioms for the "propagation" of tests into $(\!|\,|\!)$-expressions.

Let $\widehat{\Sigma}_{\mathbf{i}} := \Sigma_{\mathbf{i}} \cup \mathsf{Tests}_{\mathbf{B}}$. Let $\mathcal{B}$ be a finite list of unary and binary blocks on a finite set of labels $V$ and let $x, y \in V$. Let $b_u$ be a test for all $u \in V$. Let $\mathcal{C}$ be a finite list of metablocks (representing guarded terms in blocks) such that,

- for all unary blocks $[_u\, m\,]^{u'} \in \mathcal{B}^1$, there exists a metablock $[\!|_u\, b_u \cdot m' \cdot b_{u'}\,]\!|^{u'} \in \mathcal{C}^{\mathsf{set}}$ where $m' \in \widehat{\Sigma}_{\mathbf{i}}$;
- for all binary blocks $[_u\, c{\downarrow_z}{\uparrow^w}\, r\,]^{u'} \in \mathcal{B}^2$, the metablock $[\!|_u\, b_u{\cdot}(c{\downarrow_z}{\uparrow^w} r){\cdot}b_{u'}\,]\!|^{u'} \in \mathcal{C}^{\mathsf{set}}$ exists;
- no other metablock can be in $\mathcal{C}$.

Note that $\mathcal{C}$ is expressed with metablocks to ease the reading, but it could have been defined only with unary and binary blocks. Note also that we allowed that a unary block $[_u\, m\,]^{u'} \in \mathcal{B}^1$ can be "replaced" with one or more metablocks $[\!|_u\, b_u \cdot m' \cdot b_{u'}\,]\!|^{u'} \in \mathcal{C}^{\mathsf{set}}$ in which $m'$ is not necessarily $m$. This is to accomodate refinement hypotheses that sometimes replace an internal action by a more precise internal action when knowing if a test is true before or after the action.

There are two simple results about $\mathcal{C}$ (under the above assumptions):

$$(\!|\,\mathcal{C}\,|\!)^{y}_{x} = (\!|\,\mathcal{C}\,|\!)^{y}_{x} \cdot b_y \ , \tag{5.31}$$

$$(\!|\,\mathcal{C}\,|\!)^{y}_{x} = b_x \cdot (\!|\,\mathcal{C}\,|\!)^{y}_{x} \ . \tag{5.32}$$

*Proof.* We just prove (5.31) since the proof of (5.32) is similar.

$$\begin{aligned}
& (\!|_x\, \mathcal{C}\,|\!)^y \\
= \quad & \{\!\!\{ \text{ Backward version of (5.24) } \}\!\!\} \\
& (\textstyle\sum\, m \mid [_x\, m\,]^y \in \mathcal{C}^{\mathsf{set}} : \mathsf{mb_vpre}_{\theta_{\mathcal{C}}}(m)) \\
& + (\textstyle\sum\, m, v \mid [_v\, m\,]^y \in \mathcal{C}^{\mathsf{set}} : (\!|_x\, \mathcal{C}\,|\!)^v \cdot \mathsf{mb_vpre}_{\theta_{\mathcal{C}}}(m))
\end{aligned}$$

[2] A recursive block is a binary block that refers directly or indirectly to its starting label in the calling label.

$$
\begin{aligned}
&= && \{\!\{ \text{Hypotheses: Any metablock of } \mathcal{C} \text{ ending with } y \text{ has the test } b_y \text{ as its last element. \& Idempotency of tests \& Definition of } \mathsf{mb\_vpre}_{\theta_\mathcal{C}} \}\!\} \\
&\quad (\sum m \mid [_x\, m\,]^y \in \mathcal{C}^{\mathsf{set}} : \mathsf{mb\_vpre}_{\theta_\mathcal{C}}(m) \cdot b_y) \\
&\quad + (\sum m, v \mid [_v\, m\,]^y \in \mathcal{C}^{\mathsf{set}} : (\!|_x\, \mathcal{C}\, |\!)^v \cdot \mathsf{mb\_vpre}_{\theta_\mathcal{C}}(m) \cdot b_y) \\
&= && \{\!\{ \text{Distributivity of } \cdot \text{ over } + \}\!\} \\
&\quad \Big((\sum m \mid [_x\, m\,]^y \in \mathcal{C}^{\mathsf{set}} : \mathsf{mb\_vpre}_{\theta_\mathcal{C}}(m)) \\
&\quad + (\sum m, v \mid [_v\, m\,]^y \in \mathcal{C}^{\mathsf{set}} : (\!|_x\, \mathcal{C}\, |\!)^v \cdot \mathsf{mb\_vpre}_{\theta_\mathcal{C}}(m))\Big) \cdot b_y \\
&= && \{\!\{ \text{Backward version of (5.24)} \}\!\} \\
&\quad (\!|_x\, \mathcal{C}\, |\!)^y \cdot b_y && \blacksquare
\end{aligned}
$$

We are now ready to introduce the two new axioms for visibly pushdown Kleene algebra with tests. We add the following axioms to VPKA with tests (respectively called the *forward propagation of tests* and the *backward propagation of tests*):

$$
\begin{aligned}
&\Big(\wedge\ u, u', m \mid [\,{}^{u'}_{u}\, m\,] \in \mathcal{B}^1 : \\
&\qquad b_u \cdot m \leqslant (\sum m' \mid [\,{}^{u'}_{u}\, b_u \cdot m' \cdot b_{u'}\,] \in \mathcal{C}^{\mathsf{set}} \wedge m' \in \widehat{\Sigma}_{\mathbf{i}} : m') \cdot b_{u'}\Big) \\
&\wedge \Big(\wedge\ u, u', c, z, r, w \mid [\,{}^{w\ \ u'}_{u\ z}\, c\!\downarrow\!\uparrow r\,] \in \mathcal{B}^2 : \\
&\qquad b_u \cdot c \leqslant c \cdot b_z \quad \wedge \quad b_w \cdot r \leqslant r \cdot b_{u'}\Big) \\
&\to b_x \cdot (\!|\, \mathcal{B}\, |\!)^y_x \leqslant (\!|\, \mathcal{C}\, |\!)^y_x\ ,
\end{aligned}
\tag{5.33}
$$

$$
\begin{aligned}
&\Big(\wedge\ u, u', m \mid [\,{}^{u'}_{u}\, m\,] \in \mathcal{B}^1 : \\
&\qquad m \cdot b_{u'} \leqslant b_u \cdot (\sum m' \mid [\,{}^{u'}_{u}\, b_u \cdot m' \cdot b_{u'}\,] \in \mathcal{C}^{\mathsf{set}} \wedge m' \in \widehat{\Sigma}_{\mathbf{i}} : m')\Big) \\
&\wedge \Big(\wedge\ u, u', c, z, r, w \mid [\,{}^{w\ \ u'}_{u\ z}\, c\!\downarrow\!\uparrow r\,] \in \mathcal{B}^2 : \\
&\qquad c \cdot b_z \leqslant b_u \cdot c \quad \wedge \quad r \cdot b_{u'} \leqslant b_w \cdot r\Big) \\
&\to (\!|\, \mathcal{B}\, |\!)^y_x \cdot b_y \leqslant (\!|\, \mathcal{C}\, |\!)^y_x\ .
\end{aligned}
\tag{5.34}
$$

These axioms are valid for the language-theoretic model of VPKAT as shown in the following proof. Note that we just prove that axiom (5.33) is valid since the proof that axiom (5.34) is valid is similar.

*Proof.* Before doing the proof, we first recall an easy theorem of Kleene algebra with tests that will be useful in the proof. For all tests $b$ and $c$, and for all $p$ and $q$,

$$bp \leqslant qc \leftrightarrow bp \leqslant bqc \ . \tag{5.35}$$

We now show that axiom (5.33) is valid for the language-theoretic model of VPKAT under its natural interpretation $\mathsf{G}$. To do the proof, we use the block-based definition of the infinite family of operators for VPKAT. By definition of the language-theoretic model, we suppose that for all $u, u' \in V$ and $m \in \widehat{\Sigma}_{\mathbf{i}}$ such that $[_u\, m\, ]^{u'} \in \mathcal{B}^1$,

$$\mathsf{G}(b_u) \diamond \mathsf{G}(m) \subseteq (\cup\, m' \mid \underset{u}{[}\, b_u \cdot m' \cdot b_{u'} \overset{u'}{]} \in \mathcal{C}^{\mathsf{set}} \wedge m' \in \widehat{\Sigma}_{\mathbf{i}} : \mathsf{G}(m')) \diamond \mathsf{G}(b_{u'}) \ . \tag{5.36}$$

Also, we suppose that for all $u, u', z, w \in V$, $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$ such that $[_u\, c{\downarrow_z}{\uparrow^w}\, r\, ]^{u'} \in \mathcal{B}^2$,

$$\mathsf{G}(b_u) \diamond \mathsf{G}(c) \subseteq \mathsf{G}(c) \diamond \mathsf{G}(b_z) \ , \tag{5.37}$$

$$\mathsf{G}(b_w) \diamond \mathsf{G}(r) \subseteq \mathsf{G}(r) \diamond \mathsf{G}(b_{u'}) \ . \tag{5.38}$$

We show that

$$\mathsf{G}(b_x) \diamond \underset{x}{(\!|}\, \mathsf{G}_{\mathsf{B}}(\mathcal{B})\, \overset{y}{|\!)} \subseteq \underset{x}{(\!|}\, \mathsf{G}_{\mathsf{B}}(\mathcal{C})\, \overset{y}{|\!)} \ .$$

By definition of $(\!|_x\, \mathsf{G}_{\mathsf{B}}(\mathcal{B})\, |\!)^y$, by distributivity of $\diamond$ over $\cup$ and by set theory, it suffices to prove that for all $n \in \mathbb{N}$,

$$\mathsf{G}(b_x) \diamond \underset{x}{(\!|}\, \mathsf{G}_{\mathsf{B}}(\mathcal{B})\, \overset{y}{\underset{n}{|\!)}} \subseteq \underset{x}{(\!|}\, \mathsf{G}_{\mathsf{B}}(\mathcal{C})\, \overset{y}{|\!)} \ .$$

The proof is done by generalized induction over $n$. For the base case ($n = 0$), we show that

$$\mathsf{G}(b_x) \diamond \underset{x}{(\!|}\, \mathsf{G}_{\mathsf{B}}(\mathcal{B})\, \overset{y}{\underset{0}{|\!)}} \subseteq \underset{x}{(\!|}\, \mathsf{G}_{\mathsf{B}}(\mathcal{C})\, \overset{y}{|\!)} \ .$$

By definition of $(\!|_x\, \mathsf{G}_{\mathsf{B}}(\mathcal{B})\, |\!)^y_0$, distributivity of $\diamond$ over $\cup$ and set theory, it suffices to prove independently that, for all unary blocks $[_x\, m\, ]^y \in (\mathsf{G}_{\mathsf{B}}(\mathcal{B}))^1$,

$$\mathsf{G}(b_x) \diamond \{m\} \subseteq \underset{x}{(\!|}\, \mathsf{G}_{\mathsf{B}}(\mathcal{C})\, \overset{y}{|\!)} \ .$$

We prove it. First, by definition of $\mathsf{G}_{\mathsf{B}}$, there exists a unary block $[_x\, m_2\, ]^y$ in $\mathcal{B}^1$ that created $[_x\, m\, ]^y$. Note that, by definition of $\mathsf{G}_{\mathsf{B}}$ and $\mathsf{G}$, $m \in \mathsf{G}(m_2)$. Now,

$$
\begin{array}{ll}
& \mathsf{G}(b_x) \diamond \{m\} \\
\subseteq & \quad \{\!\{ \text{ Previous reasoning: } m \in \mathsf{G}(m_2) \text{ \& Set theory } \}\!\} \\
& \mathsf{G}(b_x) \diamond \mathsf{G}(m_2) \\
\subseteq & \quad \{\!\{ \text{ Hypothesis (5.36) \& Formula (5.35) } \}\!\} \\
& \mathsf{G}(b_x) \diamond (\cup\, m' \mid [\!]_x\, b_x \cdot m' \cdot b_y\, [\!]^y \in \mathcal{C}^{\mathsf{set}} \wedge m' \in \widehat{\Sigma}_{\mathbf{i}} : \mathsf{G}(m')) \diamond \mathsf{G}(b_y) \\
= & \quad \{\!\{ \text{ Distributivity of } \diamond \text{ on } \cup \}\!\} \\
& (\cup\, m' \mid [\!]_x\, b_x \cdot m' \cdot b_y\, [\!]^y \in \mathcal{C}^{\mathsf{set}} \wedge m' \in \widehat{\Sigma}_{\mathbf{i}} : \mathsf{G}(b_x) \diamond \mathsf{G}(m') \diamond \mathsf{G}(b_y)) \\
= & \quad \{\!\{ \text{ Interpretation } \mathsf{G} \}\!\} \\
& (\cup\, m' \mid [\!]_x\, b_x \cdot m' \cdot b_y\, [\!]^y \in \mathcal{C}^{\mathsf{set}} \wedge m' \in \widehat{\Sigma}_{\mathbf{i}} : \mathsf{G}(b_x \cdot m' \cdot b_y)) \\
\subseteq & \quad \{\!\{ \text{ Since } [\!]_x\, b_x \cdot m' \cdot b_y\, [\!]^y \in \mathcal{C}^{\mathsf{set}}, \text{ inequation (5.1) can be used \& Monotonicity} \\
& \qquad \text{of } \cup \text{ \& Definition of } \mathsf{G} \text{ and } \mathsf{G}_{\mathsf{B}} \}\!\} \\
& (\cup\, m' \mid [\!]_x\, b_x \cdot m' \cdot b_y\, [\!]^y \in \mathcal{C}^{\mathsf{set}} \wedge m' \in \widehat{\Sigma}_{\mathbf{i}} : (\!|_x\, \mathsf{G}_{\mathsf{B}}(\mathcal{C})\, |\!)^y) \\
\subseteq & \quad \{\!\{ \text{ Idempotency of } \cup \}\!\} \\
& (\!|_x\, \mathsf{G}_{\mathsf{B}}(\mathcal{C})\, |\!)^y \ .
\end{array}
$$

For the inductive case, we suppose that

$$\mathsf{G}(b_x) \diamond (\!|_x\, \mathsf{G}_{\mathsf{B}}(\mathcal{B})\, |\!)^y_k \subseteq (\!|_x\, \mathsf{G}_{\mathsf{B}}(\mathcal{C})\, |\!)^y$$

is true for all $k \in \{0, \ldots, n\}$ and for all $x, y \in V$ and we show that

$$\mathsf{G}(b_x) \diamond (\!|_x\, \mathsf{G}_{\mathsf{B}}(\mathcal{B})\, |\!)^y_{n+1} \subseteq (\!|_x\, \mathsf{G}_{\mathsf{B}}(\mathcal{C})\, |\!)^y$$

is also true. By definition of $(\!|_x\, \mathsf{G}_{\mathsf{B}}(\mathcal{B})\, |\!)^y_{n+1}$, by distributivity of $\diamond$ over $\cup$ and by set theory, it suffices to show independently that,

- for all $v \in V$ and unary blocks $[\!]_x\, m\, [\!]^v \in (\mathsf{G}_{\mathsf{B}}(\mathcal{B}))^1$,

$$\mathsf{G}(b_x) \diamond \{m\} \diamond (\!|_v\, \mathsf{G}_{\mathsf{B}}(\mathcal{B})\, |\!)^y_n \subseteq (\!|_x\, \mathsf{G}_{\mathsf{B}}(\mathcal{C})\, |\!)^y \ , \tag{5.39}$$

- for all binary blocks $[\!]_x\, \alpha_0\, c\, \alpha_1 \downarrow_z \uparrow^w \alpha_2\, r\, \alpha_3\, [\!]^y \in (\mathsf{G}_{\mathsf{B}}(\mathcal{B}))^2$,

$$\mathsf{G}(b_x) \diamond \{\alpha_0\, c\, \alpha_1\} \diamond (\!|_z\, \mathsf{G}_{\mathsf{B}}(\mathcal{B})\, |\!)^w_n \diamond \{\alpha_2\, r\, \alpha_3\} \subseteq (\!|_x\, \mathsf{G}_{\mathsf{B}}(\mathcal{C})\, |\!)^y \ , \tag{5.40}$$

- for all $v \in V$, binary blocks $[\!]_x\, \alpha_0\, c\, \alpha_1 \downarrow_z \uparrow^w \alpha_2\, r\, \alpha_3\, [\!]^v \in (\mathsf{G}_{\mathsf{B}}(\mathcal{B}))^2$ and $n_1, n_2 \in \mathbb{N}$ such that $n_1 + n_2 = n - 1$,

$$\mathsf{G}(b_x) \diamond \{\alpha_0\, c\, \alpha_1\} \diamond (\!|_z\, \mathsf{G}_{\mathsf{B}}(\mathcal{B})\, |\!)^w_{n_1} \diamond \{\alpha_2\, r\, \alpha_3\} \diamond (\!|_v\, \mathsf{G}_{\mathsf{B}}(\mathcal{B})\, |\!)^y_{n_2} \subseteq (\!|_x\, \mathsf{G}_{\mathsf{B}}(\mathcal{C})\, |\!)^y \ . \tag{5.41}$$

For (5.39), by definition of $\mathsf{G}_\mathsf{B}$, there exists a unary block $[\!]_x\, m_2 \,]\!]^v$ in $\mathcal{B}^1$ that created $[\!]_x\, m \,]\!]^v$. Note that, by definition of $\mathsf{G}_\mathsf{B}$ and $\mathsf{G}$, $m \in \mathsf{G}(m_2)$. Now,

$$
\begin{array}{ll}
 & \mathsf{G}(b_x) \diamond \{m\} \diamond (\!|_v\, \mathsf{G}_\mathsf{B}(\mathcal{B})\, |\!)^y_n \\
\subseteq & \qquad \{\!\{ \text{Previous reasoning: } m \in \mathsf{G}(m_2) \text{ \& Monotonicity of } \diamond \,\}\!\} \\
 & \mathsf{G}(b_x) \diamond \mathsf{G}(m_2) \diamond (\!|_v\, \mathsf{G}_\mathsf{B}(\mathcal{B})\, |\!)^y_n \\
\subseteq & \qquad \{\!\{ \text{Hypothesis (5.36) \& Formula (5.35) \& Monotonicity of } \diamond \,\}\!\} \\
 & \mathsf{G}(b_x) \diamond (\cup\, m' \mid [\!]_x\, b_x \cdot m' \cdot b_v \,]\!]^v \in \mathcal{C}^{\mathsf{set}} \wedge m' \in \widehat{\Sigma}_\mathbf{i} : \mathsf{G}(m')) \diamond \mathsf{G}(b_v) \diamond (\!|_v\, \mathsf{G}_\mathsf{B}(\mathcal{B})\, |\!)^y_n \\
= & \qquad \{\!\{ \text{Idempotency of tests \& Distributivity of } \diamond \text{ over } \cup \,\}\!\} \\
 & (\cup\, m' \mid [\!]_x\, b_x \cdot m' \cdot b_v \,]\!]^v \in \mathcal{C}^{\mathsf{set}} \wedge m' \in \widehat{\Sigma}_\mathbf{i} : \mathsf{G}(b_x) \diamond \mathsf{G}(m') \diamond \mathsf{G}(b_v)) \diamond \mathsf{G}(b_v) \diamond (\!|_v\, \mathsf{G}_\mathsf{B}(\mathcal{B})\, |\!)^y_n \\
\subseteq & \qquad \{\!\{ \text{Induction hypothesis \& Monotonicity of } \diamond \,\}\!\} \\
 & (\cup\, m' \mid [\!]_x\, b_x \cdot m' \cdot b_v \,]\!]^v \in \mathcal{C}^{\mathsf{set}} \wedge m' \in \widehat{\Sigma}_\mathbf{i} : \mathsf{G}(b_x) \diamond \mathsf{G}(m') \diamond \mathsf{G}(b_v)) \diamond (\!|_v\, \mathsf{G}_\mathsf{B}(\mathcal{C})\, |\!)^y \\
= & \qquad \{\!\{ \text{Interpretation } \mathsf{G} \,\}\!\} \\
 & (\cup\, m' \mid [\!]_x\, b_x \cdot m' \cdot b_v \,]\!]^v \in \mathcal{C}^{\mathsf{set}} \wedge m' \in \widehat{\Sigma}_\mathbf{i} : \mathsf{G}(b_x \cdot m' \cdot b_v)) \diamond (\!|_v\, \mathsf{G}_\mathsf{B}(\mathcal{C})\, |\!)^y \\
\subseteq & \qquad \{\!\{ \text{Since } [\!]_x\, b_x \cdot m' \cdot b_v \,]\!]^v \in \mathcal{C}^{\mathsf{set}}, \text{ inequation (5.1) can be used \& Monotonicity} \\
 & \qquad\quad \text{of } \cup \text{ and } \diamond \text{ \& Definition of } \mathsf{G} \text{ and } \mathsf{G}_\mathsf{B} \,\}\!\} \\
 & (\cup\, m' \mid [\!]_x\, b_x \cdot m' \cdot b_v \,]\!]^v \in \mathcal{C}^{\mathsf{set}} \wedge m' \in \widehat{\Sigma}_\mathbf{i} : (\!|_x\, \mathsf{G}_\mathsf{B}(\mathcal{C})\, |\!)^v) \diamond (\!|_v\, \mathsf{G}_\mathsf{B}(\mathcal{C})\, |\!)^y \\
\subseteq & \qquad \{\!\{ \text{Idempotency of } \cup \,\}\!\} \\
 & (\!|_x\, \mathsf{G}_\mathsf{B}(\mathcal{C})\, |\!)^v \diamond (\!|_v\, \mathsf{G}_\mathsf{B}(\mathcal{C})\, |\!)^y \\
\subseteq & \qquad \{\!\{ \text{Axiom (3.4) extended to metablocks } \,\}\!\} \\
 & (\!|_x\, \mathsf{G}_\mathsf{B}(\mathcal{C})\, |\!)^y \ .
\end{array}
$$

For (5.40) and (5.41), we first prove a simple lemma: for all $v \in V$, binary blocks $[\!]_x\, \alpha_0\, c\, \alpha_1 \downarrow_z \uparrow^w \alpha_2\, r\, \alpha_3 \,]\!]^v \in (\mathsf{G}_\mathsf{B}(\mathcal{B}))^2$ and $k \in \mathbb{N}$ such that $k \leqslant n$,

$$
\mathsf{G}(b_x) \diamond \{\alpha_0\, c\, \alpha_1\} \diamond \underset{z}{(\!|}\, \mathsf{G}_\mathsf{B}(\mathcal{B})\, \underset{k}{|\!)^w} \diamond \{\alpha_2\, r\, \alpha_3\} \subseteq \underset{x}{(\!|}\, \mathsf{G}_\mathsf{B}(\mathcal{C})\, \underset{}{|\!)^v} \quad . \tag{5.42}
$$

For (5.42), by definition of $\mathsf{G}_\mathsf{B}$, there exists a binary block $[\!]_x\, c' \downarrow_z \uparrow^w r' \,]\!]^v$ in $\mathcal{B}^2$ that created $[\!]_x\, \alpha_0\, c\, \alpha_1 \downarrow_z \uparrow^w \alpha_2\, r\, \alpha_3 \,]\!]^v$. Note that, by definition of $\mathsf{G}_\mathsf{B}$ and $\mathsf{G}$, $\alpha_0\, c\, \alpha_1 \in \mathsf{G}(c')$ and $\alpha_2\, r\, \alpha_3 \in \mathsf{G}(r')$. Now,

$$
\begin{array}{ll}
 & \mathsf{G}(b_x) \diamond \{\alpha_0\, c\, \alpha_1\} \diamond (\!|_z\, \mathsf{G}_\mathsf{B}(\mathcal{B})\, |\!)^w_k \diamond \{\alpha_2\, r\, \alpha_3\} \\
\subseteq & \qquad \{\!\{ \text{Previous reasonings: } \alpha_0\, c\, \alpha_1 \in \mathsf{G}(c') \text{ and } \alpha_2\, r\, \alpha_3 \in \mathsf{G}(r') \text{ \& Monotonicity} \\
 & \qquad\quad \text{of } \diamond \,\}\!\} \\
 & \mathsf{G}(b_x) \diamond \mathsf{G}(c') \diamond (\!|_z\, \mathsf{G}_\mathsf{B}(\mathcal{B})\, |\!)^w_k \diamond \mathsf{G}(r') \\
\subseteq & \qquad \{\!\{ \text{Hypothesis (5.37) \& Formula (5.35) \& Monotonicity of } \diamond \,\}\!\} \\
 & \mathsf{G}(b_x) \diamond \mathsf{G}(c') \diamond \mathsf{G}(b_z) \diamond (\!|_z\, \mathsf{G}_\mathsf{B}(\mathcal{B})\, |\!)^w_k \diamond \mathsf{G}(r')
\end{array}
$$

$$
\begin{array}{ll}
\subseteq & \{\!\{ \text{ Induction hypothesis \& Monotonicity of } \diamond \ \}\!\} \\
& \mathsf{G}(b_x) \diamond \mathsf{G}(c') \diamond (\!|_z \, \mathsf{G}_{\mathsf{B}}(\mathcal{C}) \, |\!)^w \diamond \mathsf{G}(r') \\
= & \{\!\{ \text{ Equation (5.31) } \}\!\} \\
& \mathsf{G}(b_x) \diamond \mathsf{G}(c') \diamond (\!|_z \, \mathsf{G}_{\mathsf{B}}(\mathcal{C}) \, |\!)^w \diamond \mathsf{G}(b_w) \diamond \mathsf{G}(r') \\
\subseteq & \{\!\{ \text{ Hypothesis (5.38) \& Monotonicity of } \diamond \ \}\!\} \\
& \mathsf{G}(b_x) \diamond \mathsf{G}(c') \diamond (\!|_z \, \mathsf{G}_{\mathsf{B}}(\mathcal{C}) \, |\!)^w \diamond \mathsf{G}(r') \diamond \mathsf{G}(b_v) \\
= & \{\!\{ \text{ Definition of } \mathsf{G} \text{ and } \mathsf{G}_{\mathsf{B}} \ \}\!\} \\
& \mathsf{G}(b_x \cdot c' \cdot (\!|_z \, \mathcal{C} \, |\!)^w \cdot r' \cdot b_v) \\
\subseteq & \{\!\{ \text{ Definition of } \mathcal{C} \text{: the metablock } [\!|_x \, b_x \cdot (c' \downarrow_z \uparrow^w r') \cdot b_v \, |\!]^v \in \mathcal{C}^{\mathsf{set}} \text{ exists because} \\
& \quad [\!|_x \, c' \downarrow_z \uparrow^w r' \, |\!]^v \in \mathcal{B}^2. \text{ So, inequation (5.1) can be used. \& Definition of } \mathsf{G} \\
& \quad \text{and } \mathsf{G}_{\mathsf{B}} \ \}\!\} \\
& (\!|_x \, \mathsf{G}_{\mathsf{B}}(\mathcal{C}) \, |\!)^v \ .
\end{array}
$$

Now, the proof of (5.40) is direct from (5.42) and appropriate substitution.

For (5.41),

$$
\begin{array}{ll}
& \mathsf{G}(b_x) \diamond \{\alpha_0 \, c \, \alpha_1\} \diamond (\!|_z \, \mathsf{G}_{\mathsf{B}}(\mathcal{B}) \, |\!)^w_{n_1} \diamond \{\alpha_2 \, r \, \alpha_3\} \diamond (\!|_v \, \mathsf{G}_{\mathsf{B}}(\mathcal{B}) \, |\!)^y_{n_2} \\
\subseteq & \{\!\{ \text{ Inclusion (5.42) \& Monotonicity of } \diamond \ \}\!\} \\
& (\!|_x \, \mathsf{G}_{\mathsf{B}}(\mathcal{C}) \, |\!)^v \diamond (\!|_v \, \mathsf{G}_{\mathsf{B}}(\mathcal{B}) \, |\!)^y_{n_2} \\
= & \{\!\{ \text{ Equation (5.31) } \}\!\} \\
& (\!|_x \, \mathsf{G}_{\mathsf{B}}(\mathcal{C}) \, |\!)^v \diamond \mathsf{G}(b_v) \diamond (\!|_v \, \mathsf{G}_{\mathsf{B}}(\mathcal{B}) \, |\!)^y_{n_2} \\
\subseteq & \{\!\{ \text{ Induction hypothesis \& Monotonicity of } \diamond \ \}\!\} \\
& (\!|_x \, \mathsf{G}_{\mathsf{B}}(\mathcal{C}) \, |\!)^v \diamond (\!|_v \, \mathsf{G}_{\mathsf{B}}(\mathcal{C}) \, |\!)^y \\
\subseteq & \{\!\{ \text{ Axiom (3.4) extended to metablocks } \}\!\} \\
& (\!|_x \, \mathsf{G}_{\mathsf{B}}(\mathcal{C}) \, |\!)^y \ .
\end{array}
$$

■

Note that when $\mathcal{C}$ is such that for all unary blocks $[\!|_u \, m \, |\!]^{u'} \in \mathcal{B}^1$, the metablock $[\!|_u \, b_u \cdot m \cdot b_{u'} \, |\!]^{u'} \in \mathcal{C}^{\mathsf{set}}$ exists, then the two following laws are easily derivable for visibly pushdown Kleene algebra with tests.

$$
\boxed{
\begin{array}{l}
(\wedge \, u, u', m \mid [\underset{u}{|} \, m \, \overset{u'}{|}] \in \mathcal{B}^1 : b_u \cdot m \leqslant m \cdot b_{u'}) \\
\wedge \, (\wedge \, u, u', c, z, r, w \mid [\underset{u}{|} \, c \underset{z}{\downarrow} \overset{w}{\uparrow} r \, \overset{u'}{|}] \in \mathcal{B}^2 : b_u \cdot c \leqslant c \cdot b_z \ \wedge \ b_w \cdot r \leqslant r \cdot b_{u'}) \\
\rightarrow b_x \cdot (\!|\underset{x}{\mathcal{B}}\overset{y}{}|\!) \leqslant (\!|\underset{x}{\mathcal{C}}\overset{y}{}|\!) \ ,
\end{array}
} \tag{5.43}
$$

$$
\begin{array}{l}
\quad (\wedge\ u, u', m \mid [\underset{u}{\ } m \overset{u'}{\ }] \in \mathcal{B}^1 : m \cdot b_{u'} \leqslant b_u \cdot m) \\
\wedge\ (\wedge\ u, u', c, z, r, w \mid [\underset{u}{\ } c \underset{z}{\downarrow} \overset{w}{\uparrow} r \overset{u'}{\ }] \in \mathcal{B}^2 : c \cdot b_z \leqslant b_u \cdot c\ \wedge\ r \cdot b_{u'} \leqslant b_w \cdot r) \\
\rightarrow (\!|\, \mathcal{B} \,|\!)^{y}_{x} \cdot b_y \leqslant (\!|\, \mathcal{C} \,|\!)^{y}_{x}\ .
\end{array}
\tag{5.44}
$$

# Chapter 6

# Interprocedural Program Analysis Using VPKAT

This chapter shows that the proposed formalism called visibly pushdown Kleene algebra extended with tests can be used for doing some interprocedural program analyses like formal verification and verification of compiler optimizations.

## 6.1 VPKAT as a Modelling Tool: Encoding an Interprocedural Program

VPKAT can be used to encode an interprocedural program written in an imperative programming language like C. This section extends the "classical" way of encoding programs in Kleene-like algebras [6, 27].

The goal is to encode a program (and its associated property or optimization) in VPKAT. In other words, a formula written in VPKAT should be "extracted" from the program (and its associated property or optimization). To do this, the encoding is done in three steps. The first step is obvious, one should define the desired abstraction for atomic program instructions and variables through the sets $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$, $\Sigma_{\mathbf{r}}$ and $\mathbf{B}$. After all, to create a formula of VPKAT, one first needs the alphabets for the atomic elements. Usually, a crude abstraction is taken for the atomic instructions: each instruction of the program is considered an internal action. The desired abstraction for the variables is more complex. They are usually represented by a set of atomic tests. This set varies a lot depending on the type of the variable and of the verification to be done. For

example, it may be sufficient to encode an integer variable $n$ used in a program by using only a test like "is $n$ equal to 0 ?". However, it can be as complex as trying to represent lots of tests for $n$ by Booleans (similar to the binary encoding of an integer by a compiler). For more flexibility, the encoding of atomic instructions and variables is left at the user's discretion. As a result, this abstraction step is semi-automatic. Lots of works already done in software verification can help to guide this abstraction step (see, for example, [3]).

The first step of the encoding takes care of representing the *atomic* elements of a program. So, it remains to take care of the standard programming constructs (including functions) to give an expression of VPKAT. This is the goal of the second step. The second step encodes the standard programming constructs and gives an expression of MBexp:

$$s;t := s \cdot t, \qquad \textbf{if } b \textbf{ then } s \textbf{ else } t := b \cdot s + \overline{b} \cdot t, \qquad \textbf{while } b \textbf{ do } s := (b \cdot s)^* \cdot \overline{b},$$

where $b$ is a test and $s$ and $t$ are programs, whereas any function $f$ gives a metablock $[\![_f\, s\,]\!]^\tau$ where $s$ is the body of the function and $\tau$ is a label used to indicate the end of the body of any function. The whole expression for the program is an expression $(\!|_{x'}\, [\!]_{x'}\, \langle x \downarrow_x \uparrow^\tau x \rangle\, ]\!]^\tau, \mathcal{B}\, |\!)^\tau$ where $\mathcal{B}$ represents the encoding of all the individual functions of the program and $x$ is the label used for the designated main function of the program. Note that the main function of the program (here, $x$) is enclosed by its proper call action and return action just like any other function by having $x'$ has the starting label of the expression. This is why the binary block $[\!]_{x'}\, \langle x \downarrow_x \uparrow^\tau x \rangle\, ]\!]^\tau$ have been added to the expression.

To further ease the use of functions, the following abbreviation is used when "calling" functions in VPKA: ${}^{\langle c} z^{r\rangle} := (\langle c \downarrow_z \uparrow^\tau r\rangle)$ for a call action $\langle c$, a return action $r\rangle$ and a label $z$. Also, two abbreviations of the abbreviation are used:

- ${}^{\langle} z^{\rangle} := {}^{\langle z} z^{z\rangle}$ for a label $z$;
- ${}^{\langle_i} z^{i\rangle} := {}^{\langle z_i} z^{z_i\rangle}$ for a label $z$.

These abbreviations allow us to put an emphasis on the name of the function being called instead of the associated call action and return action. We hope that it will be more natural for a programmer.

The encoding of the programming constructs presented previously is very close to

```
void f() {
  p;
  while (b) {
    q;
  }
}
```

Figure 6.1: Abstract program containing only one non-recursive function.

the syntax of the programming language. This should highlight the analogy with programming languages. In particular, the sequence operator is well represented by the concatenation operator, the conditional instruction is well represented by a choice between two options guarded respectively by the test and by the complementation of the test, and the while loop is *almost* well represented by the $^*$ operator and the guard which states that the body of the loop is executed until the test is false. The term *almost* is used to emphasize that the $^*$ operator only deals with finite loops, not infinite loops.

Some examples of the second step are presented. In these examples, consider that the source code of the programs already contains "abstracted" elements of the algebra (the result of the first step). As a first example, let us consider the abstract program of Figure 6.1 containing only one non-recursive function that is called $f$. The result of the second step produces the following expression:

$$\underset{f'}{(\!|} \left[\!\!\left[ \underset{f'}{\langle f \rangle} \right]\!\!\right]^{\tau}, \left[\!\!\left[ \underset{f}{p \cdot (b \cdot q)^* \cdot \overline{b}} \right]\!\!\right]^{\tau} \underset{f'}{|\!)}^{\tau} \ .$$

The resulting expression matches the abstracted program. We clearly see the single function and the structure of the loop.

As a second example, let us consider the abstract program of Figure 6.2 containing only two non-recursive functions that are called $f$ and $g$. Note that $f$ is exactly the same function as in the previous example. Consider that $g$ is the main function of the program. The result of the second step produces the following expression:

$$\underset{g'}{(\!|} \left[\!\!\left[ \underset{g'}{\langle g \rangle} \right]\!\!\right]^{\tau}, \left[\!\!\left[ \underset{f}{p \cdot (b \cdot q)^* \cdot \overline{b}} \right]\!\!\right]^{\tau}, \left[\!\!\left[ \underset{g}{r \cdot \langle f \rangle \cdot s} \right]\!\!\right]^{\tau} \underset{g'}{|\!)}^{\tau} \ .$$

Once again, the resulting expression matches the abstracted program. We clearly see the two functions and the call of the function $f$. Note that the abbreviation used to call the function $f$ in $g$ also hides the complexity of using the call action $\langle f$ and the return action $f \rangle$.

```
void f() {
  p;
  while (b) {
    q;
  }
}

void g() {
  r;
  f();
  s;
}
```

Figure 6.2: Abstract program containing only two non-recursive functions.

```
void h() {
  if (b) {
    r;
  } else {
    p;
    h();
    q;
  }
}
```

Figure 6.3: Abstract program containing one recursive function.

As a third example, let us consider the abstract program of Figure 6.3 containing one recursive function that is called $h$. The result of the second step produces the following expression:

$$\Big(\!\Big| \underset{h'}{\big[} \underset{h'}{\big[} \langle h \rangle \big]^{\tau}, \underset{h}{\big[} b \cdot r + \overline{b} \cdot p \cdot \langle h \rangle \cdot q \big]^{\tau} \Big|\!\Big)^{\tau} \ .$$

Note how intuitive the recursive call is in this setting. It is the same thing as calling any other function.

As a final example, let us consider the abstract program of Figure 6.4 containing mutually recursive functions $k$ and $l$. Consider that $k$ is the main function of the program. The result of the second step produces the following expression:

$$\Big(\!\Big| \underset{k'}{\big[} \underset{k'}{\big[} \langle k \rangle \big]^{\tau}, \underset{k}{\big[} b_1 \cdot r_1 + \overline{b_1} \cdot p_1 \cdot \langle l \rangle \cdot q \big]^{\tau}, \underset{l}{\big[} b_2 \cdot r_2 + \overline{b_2} \cdot p_2 \cdot \langle k \rangle \big]^{\tau} \Big|\!\Big)^{\tau} \ .$$

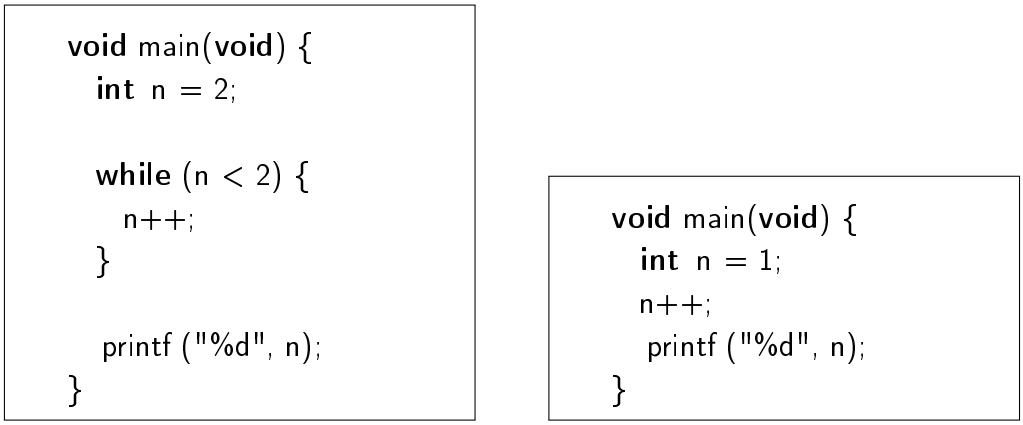
```
void k() {
  if (b1) {
    r1;
  } else {
    p1;
    l();
    q;
  }
}

void l() {
  if (b2) {
    r2;
  } else {
    p2;
    k();
  }
}
```

Figure 6.4: Abstract program containing mutually recursive functions.

Note how intuitive the mutual recursive calls are handled. It is the same thing as calling any other functions.

Recall that we said that the while loop is *almost* well represented by the $^*$ operator. In fact, in the proposed encoding, any infinite loop is "lost" in the encoding. For example, the infinite loop **while** (true) **do** p; is encoded by $(1 \cdot p) \cdot 0$ and so it is equal to 0 by Kleene algebra although there is an infinite loop here. Something similar happens when calling functions infinitely often within blocks. So, in the remainder of this chapter, infinite behaviours will be avoided (this is left as future work). Note that halting programs are obtained by restricting recursive procedures and loops to simple cases.

Incidentally, when seeing the result of the first two steps of the encoding of programs, one could ask if VPKAT is powerful enough to prove program equivalence or to verify properties of programs. This is not the case. The reason is primarily linked with the first step (and the definition of the algebra of course): an abstraction of the original program is done for the atomic elements. This abstraction allows us to do algebraic reasoning over the program, but it can lack some features of the original program necessary to prove properties about it. All in all, the second step of the encoding




(a) First program (b) Second program

Figure 6.5: Two semantically equivalent programs (when considering only inputs and outputs).

encodes the *control flow* of the program, but it misses the *data flow* of the program! In particular, the semantics of the instructions is not considered! Let us see this by an example. Consider the two programs of Figure 6.5. If we consider the semantics of the instructions, we can see that the body of the loop of the first program will never be executed. So, the first program only assigns 2 to the variable n and writes the value of n to the console. The second program is straightforward: it assigns 1 to n, increments n by 1 (thus yielding 2) and writes the value of n to the console. So, the two programs are semantically equivalent when considering *relational equivalence* (when we are only interested in inputs and outputs of the programs).

Using the first two steps of the encoding of programs in VPKAT, one can first define the following abstraction:

$$\begin{array}{lll} b & \text{represents the test} & \mathsf{n} < 2 \quad , \\ p & \text{represents the internal action} & \mathsf{n} = 2; \quad , \\ q & \text{represents the internal action} & \mathsf{n{+}{+}}; \quad , \\ r & \text{represents the internal action} & \mathsf{n} = 1; \quad , \\ s & \text{represents the internal action} & \mathsf{printf}\ (\text{"\%d"}, \mathsf{n}); \quad , \end{array}$$

and then encode the first program by

$$\Big(\!\Big|\, \underset{m'}{[}\, \underset{m'}{\langle m \rangle}^{\tau} \,\underset{m}{]}, \; \underset{m}{[}\, p \cdot (b \cdot q)^* \cdot \bar{b} \cdot s \,]^{\tau\tau} \,\Big|\!\Big)$$

and the second by

$$\Big(\!\Big|\, \underset{m'}{[}\, \underset{m'}{\langle m \rangle}^{\tau} \,\underset{m}{]}, \; \underset{m}{[}\, r \cdot q \cdot s \,]^{\tau\tau} \,\Big|\!\Big) \quad .$$

So, to verify whether the two programs are equivalent using VPKAT, we should verify whether

$$\mathbb{(}\,[\,{}^{\tau}_{m'}\langle m\rangle_{m'}\,],\ [\,{}^{\tau\,\tau}_{m}\,p\cdot(b\cdot q)^*\cdot\overline{b}\cdot s\,]\,\mathbb{)} \quad = \quad \mathbb{(}\,[\,{}^{\tau}_{m'}\langle m\rangle_{m'}\,],\ [\,{}^{\tau\,\tau}_{m}\,r\cdot q\cdot s\,]\,\mathbb{)}$$

is a theorem of VPKAT. However, this cannot be the case. Recall that equality in VPKA represents the language equivalence between visibly pushdown languages. In VPKAT, the equality now represents *trace equivalence* between Kripke structures (as inspired by [27]). The two previous expressions are not trace equivalent mainly because:

- the loop cannot be removed in the first expression ($p$ has no semantics, it is just a token saying that an action occurs, but it is not known what this instruction does);
- assigning 2 to n is a different operation than assigning 1 to n and increment it by one just after.

So, the first two steps in VPKAT yield expressions that are, most of the time, too restrictive for us. Usually, when we compare programs, we intend to be more "flexible" than trace equivalence. Fortunately, VPKAT allows us to extend its notion of equivalence at will. Thus, we can decide what we mean by "semantically equivalent" (it should be an extension of trace equivalence). The idea is to use *hypotheses* for the verification (this will be our third step of the encoding). In other words, Horn formulae will be used instead of simple equations. The hypotheses used in these formulae will represent some *semantic equivalences* that are easily verifiable. Generally, the hypotheses state simple facts about the program and usually involve only few atomic elements. This allows us to retrieve some of the semantics of the atomic elements and help to constitute a simple data flow. For example, using the previous encoding, it is correct to define a hypothesis such as

$$p = p\cdot\overline{b}$$

when we consider only inputs and outputs. It simply states that just after the execution of $p$ (n = 2;), the test $b$ (n < 2) is always false. Also, it is correct to define a hypothesis such as

$$p = r\cdot q$$

when we consider only inputs and outputs. After all, this states that assigning 2 to n is the same thing as assigning 1 to n and immediately increment it by one. Thus, the complete formula to verify is:

$$p = p\cdot\overline{b}\ \wedge\ p = r\cdot q \quad\rightarrow\quad \mathbb{(}\,[\,{}^{\tau}_{m'}\langle m\rangle_{m'}\,],\ [\,{}^{\tau\,\tau}_{m}\,p\cdot(b\cdot q)^*\cdot\overline{b}\cdot s\,]\,\mathbb{)} \quad = \quad \mathbb{(}\,[\,{}^{\tau}_{m'}\langle m\rangle_{m'}\,],\ [\,{}^{\tau\,\tau}_{m}\,r\cdot q\cdot s\,]\,\mathbb{)}\ .$$

This formula is a theorem of VPKAT. We suppose $p = p \cdot \overline{b}$ and $p = r \cdot q$, and we show

$$(\!|\ [\!]_{m'}\ \langle m \rangle^{\tau}\ [\!]_{m'},\ [\!]_{m}\ p \cdot (b \cdot q)^* \cdot \overline{b} \cdot s\ [\!]_{m}^{\tau\,\tau}\ |\!) \quad = \quad (\!|\ [\!]_{m'}\ \langle m \rangle^{\tau}\ [\!]_{m'},\ [\!]_{m}\ r \cdot q \cdot s\ [\!]_{m}^{\tau\,\tau}\ |\!) \ .$$

By law (5.25), it suffices to show that

$$p \cdot (b \cdot q)^* \cdot \overline{b} \cdot s \quad = \quad r \cdot q \cdot s \ .$$

We prove it.

$$\begin{array}{ll}
 & p \cdot (b \cdot q)^* \cdot \overline{b} \cdot s \\
= & \qquad \{\!\{ \text{ Hypothesis: } p = p \cdot \overline{b} \ \}\!\} \\
 & p \cdot \overline{b} \cdot (b \cdot q)^* \cdot \overline{b} \cdot s \\
= & \qquad \{\!\{ \text{ Kleene algebra: } t^* = 1 + t \cdot t^* \ \}\!\} \\
 & p \cdot \overline{b} \cdot (1 + b \cdot q \cdot (b \cdot q)^*) \cdot \overline{b} \cdot s \\
= & \qquad \{\!\{ \text{ Distributivity of } \cdot \text{ over } + \text{ \& Identity of } \cdot \ \}\!\} \\
 & p \cdot (\overline{b} + \overline{b} \cdot b \cdot q \cdot (b \cdot q)^*) \cdot \overline{b} \cdot s \\
= & \qquad \{\!\{ \text{ Contradiction: } \overline{b} \cdot b = 0 \text{ \& Zero of } \cdot \text{ \& Identity of } + \ \}\!\} \\
 & p \cdot \overline{b} \cdot \overline{b} \cdot s \\
= & \qquad \{\!\{ \text{ Hypothesis: } p = p \cdot \overline{b}, \text{ twice } \}\!\} \\
 & p \cdot s \\
= & \qquad \{\!\{ \text{ Hypothesis: } p = r \cdot q \ \}\!\} \\
 & r \cdot q \cdot s
\end{array}$$

In short, the third step of the encoding of a program in VPKAT is to encode the desired semantics of atomic program instructions, variables and variable passing mechanisms by a set of equational hypotheses $\mathcal{H}$. This step is powerful, but it is also the most tedious one: the hypotheses usually represent an abstraction of the data flow of the program as we saw in the previous example. However, it is also useful to have some hypotheses that represent the variable passing mechanism of the procedure. We present some of these in Section 6.2. It is also possible to encode other assumptions about the program (like some optimization assumptions that will be used in Section 6.3).

Recall that it should be easy to verify that the program satisfies the hypotheses. Some classes (patterns) of hypotheses are already studied in Kleene algebra with tests (see for example [27]). In our experiments, these patterns are used often and represent the majority of the semantic equivalences that one usually needs. Surprisingly, these patterns are most of the time equivalent to an equation of the form $p = 0$ for an expression $p$.

The most used pattern is that of hypotheses of the form $b_1 p = b_1 p b_2$. These hypotheses had a great impact in program verification in Kleene-like algebras [28, 33, 41, 50]. The reason is that these hypotheses represent Hoare's partial correctness assertion $\{b_1\}p\{b_2\}$. So, they allow one to study the behaviour of an instruction, and they are already well used in program verification in general. The idea is that if a precondition $b_1$ is satisfied before doing an instruction $p$ (and the instruction $p$ terminates), then the postcondition $b_2$ is satisfied after the instruction. Note that the hypothesis $b_1 p = b_1 p b_2$ can be represented by four equivalent forms [30]:

$$b_1 p = b_1 p b_2 \ , \qquad b_1 p \leqslant p b_2 \ , \qquad p\overline{b_2} \leqslant \overline{b_1} p \ , \qquad b_1 p \overline{b_2} = 0 \ .$$

Another usual pattern is that of hypotheses of the form $b_1 p = p b_2$. They are a generalization of hypotheses of commutativity between a test and a program ($b_1 p = p b_1$). In essence, they state that a precondition $b_1$ is satisfied just before executing $p$ if and only if a postcondition $b_2$ is satisfied just after executing $p$. As for the previous pattern, these hypotheses have several equivalent forms [6]:

$$b_1 p = p b_2 \ , \qquad \overline{b_1} p = p \overline{b_2} \ , \qquad b_1 p \overline{b_2} + \overline{b_1} p b_2 = 0 \ .$$

The last pattern of hypotheses presented here is one often used for the verification of program optimizations. They are hypotheses of the form $bp = b$ where $p$ is an atomic instruction. Intuitively, such a hypothesis states that executing $p$ when knowing that the test $b$ is true just before the execution is useless. Kozen and Hardin showed that these hypotheses do not correspond to any hypothesis of the form $q = 0$ [20]. These hypotheses will not be used in this dissertation.

In short, the encoding of an interprocedural program written in an imperative programming language like C in VPKAT is done in three steps:

1. Define the desired abstraction for atomic program instructions and variables through the sets $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$, $\Sigma_{\mathbf{r}}$ and $\mathbf{B}$;
2. Encode the program control flow by a VPRE $p$;
3. Encode the desired semantics of atomic program instructions, variables and variable passing mechanisms by a set of equational hypotheses $\mathcal{H}$. It is also possible to encode other assumptions about the program (like some optimization assumptions that will be used in Section 6.3).

Notice how elegant and clean the "classical" way of encoding programs in Kleene-like algebras is. In particular, the separation is clear between the representation of the

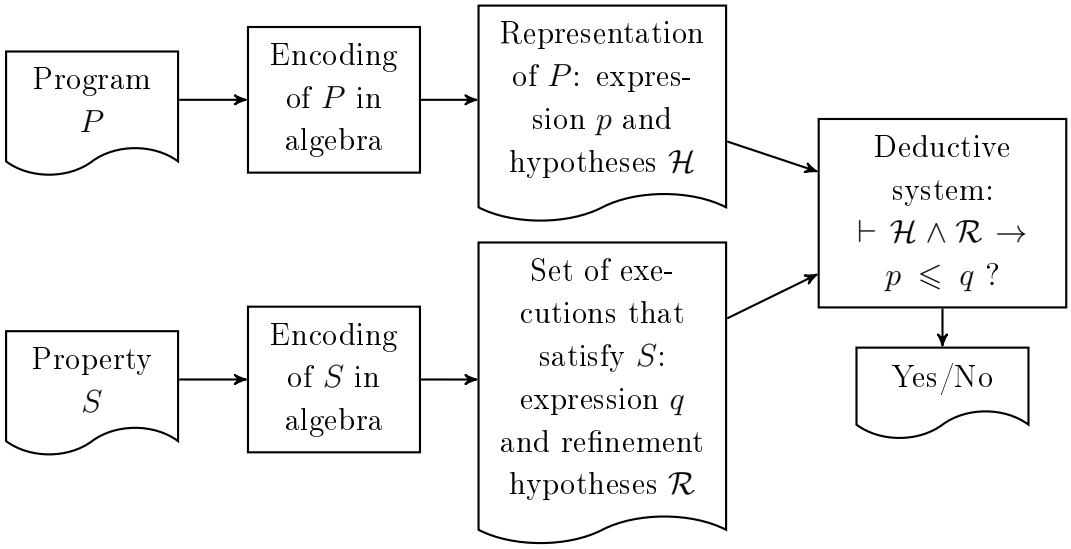



Figure 6.6: Framework of static analysis in visibly pushdown Kleene algebra with tests.

control flow of the program and the semantics used for the atomic instructions. This enables one to add and remove hypotheses at will without modifying the control flow! So, it naturally supports refinement and abstraction of the data flow without modifying the control flow! The more hypotheses are given for an equation, the more refined is the model (abstract program). Of course, it is always a challenge to find the right abstraction level for analysis.

## 6.2 Formal Verification of Interprocedural Programs

Here is a first application of VPKAT for formal verification. It is a framework for static analysis like the framework defined in [30] or in [7].

Formal verification of interprocedural programs follows the process in Figure 6.6. First, encode the program in VPKAT as we have explained in Section 6.1. The encoding gives a VPRE $p$ representing the program control flow and a set of equational hypotheses $\mathcal{H}$ representing the desired semantics of the program.

Currently, the framework deals only with halting programs, since non-halting programs need specific mechanisms (see for example [30]). Halting programs are obtained by restricting recursive procedures and loops to simple cases.

The encoding of the property in VPKAT is left at the user's discretion. Since the

framework is restricted to halting programs, the full power of visibly pushdown automata can be used to define the property (this is not the case with non-halting programs). However, defining a property by such automata does not always seem easy. In our experiments, we found it clearer to define the property directly in VPKAT. This has the drawback of not having a fixed encoding like in [7] or [30]. We used an encoding of the visibly pushdown language representing the set of executions of any program on $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$, $\Sigma_{\mathbf{r}}$ and $\mathbf{B}$ that satisfy the property; that encoding is composed of an expression $q$ of VPKAT and a set $\mathcal{R}$ of "refinement hypotheses" that help to sharpen the program abstraction according to the desired property. Some refinement hypotheses are presented in Section 6.2.1.

The static analysis ends by verifying whether $\mathcal{H} \wedge \mathcal{R} \to p \leqslant q$ is a theorem of VPKAT. The program is said to satisfy the property if and only if the formula is a theorem of VPKAT.

### 6.2.1 Example

Take the abstract program of Figure 6.7a. In it, the action $\mathsf{open}(i)$ (respectively $\mathsf{close}\,(i)$) represents opening and writing to the file named $i$ (respectively, the writing to and closing the file named $i$). To simplify the example, suppose that these actions are internal actions. We want to show the non-regular property $S$:

*Any file opened in a procedure must be closed exactly in this procedure.*

We prove it for the cases where $i \in \{0, 1\}$ (this is a drastic finite abstraction of the possible values of the variable).

For the encoding of the program, use an atomic test $\mathsf{b}$ to represent $i = 0$ and $\overline{\mathsf{b}}$ to represent $i = 1$. So, $\mathbf{B} := \{\mathsf{b}\}$. Also, use $\Sigma_{\mathbf{i}} := \{\mathsf{o}, \mathsf{c}\}$, $\Sigma_{\mathbf{c}} := \{\langle \mathsf{f}\}$, $\Sigma_{\mathbf{r}} := \{\mathsf{f}\rangle\}$ and $V := \{\mathsf{f}, \mathsf{f}', \tau\}$, where $\mathsf{f}'$ is used as the main procedure. Note that actions $\mathsf{open}(i)$ and $\mathsf{close}\,(i)$ are abstracted by actions $\mathsf{o}$ and $\mathsf{c}$. The information that they depend on the value of $i$ is lost. This will be taken care of in the refinement hypotheses. The encoding of the program control flow $p$ is given in Figure 6.7b.

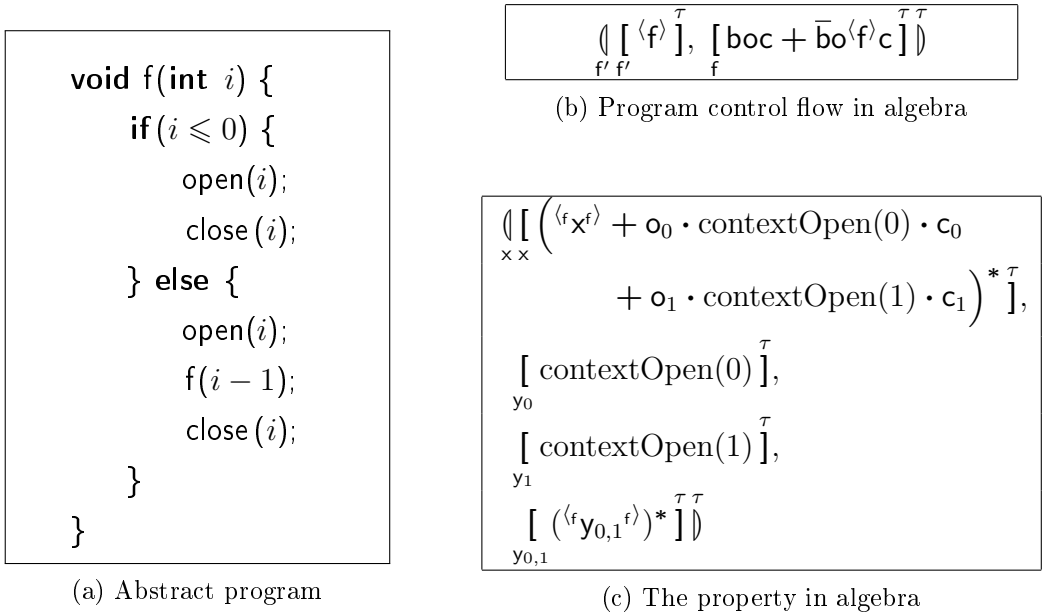



Figure 6.7: Elements for the example of interprocedural analysis.

The encoding of the desired semantics is given only for hypotheses used in the proof. So,

$$\begin{aligned}\mathcal{H} \;:=\;& \mathsf{bo} = \mathsf{ob} \\ & \wedge\, \overline{\mathsf{b}} \cdot \langle \mathsf{f} \leqslant \langle \mathsf{f} \cdot \mathsf{b} \\ & \wedge\, \overline{\mathsf{b}} \cdot \langle \mathsf{f} \cdot (\!|_{\mathsf{f}}\, \mathcal{B}\, |\!)^{\tau} \cdot \mathsf{f}\rangle = \langle \mathsf{f} \cdot (\!|_{\mathsf{f}}\, \mathcal{B}\, |\!)^{\tau} \cdot \mathsf{f}\rangle \cdot \overline{\mathsf{b}}\end{aligned}$$

where

$$\mathcal{B} \;:=\; \underset{\mathsf{f}'}{[}\, \langle \mathsf{f} \rangle \,\underset{\mathsf{f}'}{]}^{\tau},\ \underset{\mathsf{f}}{[}\, \mathsf{boc} + \overline{\mathsf{b}}\mathsf{o}\langle \mathsf{f}\rangle \mathsf{c}\,\underset{\mathsf{f}}{]}^{\tau} \ .$$

The first hypothesis states that action $\mathsf{o}$ does not modify the test $\mathsf{b}$. The second and third hypotheses abstractly encode the passing of the variable $i$ by value: the second is a Hoare triple that states that if $i = 1$ just before calling $f$, then the value of $i$ at the beginning of the newly called $f$ is now 0 (since the actual argument of $f$ in the program is $i - 1$), and the third states that the value of $i$ just before calling $f$ is remembered just after returning from $f$.

As we saw earlier, the property is relative to the exact file used, but the encoding of the program control flow does not give enough information about which file is really used. To do this, add four internal actions, namely $\mathsf{o}_0$, $\mathsf{o}_1$, $\mathsf{c}_0$ and $\mathsf{c}_1$ to $\Sigma_{\mathbf{i}}$ and define

$$\begin{aligned}\mathcal{R} \;:=\;& \mathsf{bo} \leqslant \mathsf{o}_0 \\ & \wedge\, \overline{\mathsf{b}}\mathsf{o} \leqslant \mathsf{o}_1 \\ & \wedge\, \mathsf{bc} \leqslant \mathsf{c}_0 \\ & \wedge\, \overline{\mathsf{b}}\mathsf{c} \leqslant \mathsf{c}_1 \ .\end{aligned}$$

These hypotheses state that actions $\mathsf{o}$ and $\mathsf{c}$ depend on $\mathsf{b}$ and $\overline{\mathsf{b}}$.

The encoding of the property $S$ by an expression $q$ uses metablocks having starting labels $\{\mathsf{x}, \mathsf{y}_0, \mathsf{y}_1, \mathsf{y}_{0,1}\}$. Intuitively,

- the label $\mathsf{x}$ represents that no file is opened;
- each label $\mathsf{y}_j$, for $j \in \{0,1\}$, represents that only the file $j$ is opened;
- the label $\mathsf{y}_{0,1}$ represents that the two files are opened.

To ease the definition of the property, for $i \in \{0,1\}$, let

$$\begin{aligned} &\mathrm{other}(0) := 1 \ , \qquad \mathrm{other}(1) := 0 \ , \\ &\mathrm{contextOpen}(i) := \big(\langle \mathsf{f} \mathsf{y}_i{}^{\mathsf{f}} \rangle + \mathsf{o}_{\mathrm{other}(i)} (\langle \mathsf{f} \mathsf{y}_{0,1}{}^{\mathsf{f}} \rangle)^* \mathsf{c}_{\mathrm{other}(i)}\big)^* \ . \end{aligned}$$

The metablock expression $\mathrm{contextOpen}(i)$ states that, for the context in which file $i$ is already opened, a program can call function $f$ and keep the context as is, or it can open and close the other file interleaved with the call of the function $f$ with the new context that the two files are opened. Of course, this choice can be done any finite number of times. The encoding of the property $S$ by an expression $q$ is given in Figure 6.7c.

The program satisfies $S$ if $\mathcal{H} \wedge \mathcal{R} \to p \leqslant q$ is a theorem of VPKAT. Let

$$\mathcal{B} := \underset{\mathsf{f}'}{[} \langle \mathsf{f} \rangle \overset{\tau}{]}, \underset{\mathsf{f}}{[} \mathsf{boc} + \overline{\mathsf{b}} \mathsf{o} \langle \mathsf{f} \rangle \mathsf{c} \overset{\tau}{]} \ .$$

Note that no correct travelling can *start* with $\tau$. So, by (5.24),

$$\underset{\tau}{(\!|} \mathcal{B} \overset{\tau}{|\!)} = 0 \ . \tag{6.1}$$

Now, let

$$\begin{aligned} \mathcal{C} := \underset{\mathsf{x}}{[} (\langle \mathsf{f} \mathsf{x}^{\mathsf{f}} \rangle + \mathsf{o}_0 \cdot \mathrm{contextOpen}(0) \cdot \mathsf{c}_0 + \mathsf{o}_1 \cdot \mathrm{contextOpen}(1) \cdot \mathsf{c}_1)^* \overset{\tau}{]}, \\ \underset{\mathsf{y}_0}{[} \mathrm{contextOpen}(0) \overset{\tau}{]}, \underset{\mathsf{y}_1}{[} \mathrm{contextOpen}(1) \overset{\tau}{]}, \underset{\mathsf{y}_{0,1}}{[} (\langle \mathsf{f} \mathsf{y}_{0,1}{}^{\mathsf{f}} \rangle)^* \overset{\tau}{]} \ . \end{aligned}$$

We first prove that

$$\mathsf{o}_0 \mathsf{c}_0 \leqslant \underset{\mathsf{y}_1}{(\!|} \mathcal{C} \overset{\tau}{|\!)} \ . \tag{6.2}$$

By inequation (5.1) and the transitivity of $\leqslant$, it suffices to prove that

$$\mathsf{o}_0 \mathsf{c}_0 \leqslant \mathsf{mb\_vpre}_{\theta_C}(\mathrm{contextOpen}(1)) \ .$$

$$
\begin{aligned}
& \mathsf{o}_0\mathsf{c}_0 \\
= \quad & \{\!\{ \text{ Identity of } \cdot \ \}\!\} \\
& \mathsf{o}_0 \cdot 1 \cdot \mathsf{c}_0 \\
\leqslant \quad & \{\!\{ \text{ Kleene algebra: for any expression } r,\ 1 \leqslant r^* \ \&\ \text{Monotonicity of } \cdot \ \}\!\} \\
& \mathsf{o}_0 \cdot (\langle \mathsf{f} \cdot (\!|_{\mathsf{y}_{0,1}}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle)^* \cdot \mathsf{c}_0 \\
\leqslant \quad & \{\!\{ \text{ Kleene algebra: for any expression } r,\ r \leqslant r^* \ \}\!\} \\
& \Big(\mathsf{o}_0 \cdot (\langle \mathsf{f} \cdot (\!|_{\mathsf{y}_{0,1}}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle)^* \cdot \mathsf{c}_0\Big)^* \\
\leqslant \quad & \{\!\{ \text{ Kleene algebra: for any expressions } r_1 \text{ and } r_2,\ r_1^* \leqslant (r_2 + r_1)^* \ \}\!\} \\
& \Big(\langle \mathsf{f} \cdot (\!|_{\mathsf{y}_1}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle + \mathsf{o}_0 \cdot (\langle \mathsf{f} \cdot (\!|_{\mathsf{y}_{0,1}}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle)^* \cdot \mathsf{c}_0\Big)^* \\
= \quad & \{\!\{ \text{ Definition of contextOpen}(1) \text{ and } \mathsf{mb\_vpre}_{\theta_{\mathcal{C}}} \ \}\!\} \\
& \mathsf{mb\_vpre}_{\theta_{\mathcal{C}}}(\text{contextOpen}(1))
\end{aligned}
$$

Then, we prove that

$$
\mathsf{o}_0\mathsf{c}_0 + \mathsf{o}_1 \cdot \langle \mathsf{f} \cdot \underset{\mathsf{y}_1}{(\!|}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle \cdot \mathsf{c}_1 \leqslant \underset{\mathsf{x}}{(\!|}\, \mathcal{C}\, |\!)^{\tau} \ . \tag{6.3}
$$

By inequation (5.1) and the transitivity of $\leqslant$, it suffices to prove that

$$
\begin{aligned}
& \mathsf{o}_0\mathsf{c}_0 + \mathsf{o}_1 \cdot \langle \mathsf{f} \cdot (\!|_{\mathsf{y}_1}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle \cdot \mathsf{c}_1 \\
& \leqslant \ \mathsf{mb\_vpre}_{\theta_{\mathcal{C}}}\Big((\langle {}^{\mathsf{f}}\mathsf{x}^{\mathsf{f}} \rangle + \mathsf{o}_0 \cdot \text{contextOpen}(0) \cdot \mathsf{c}_0 + \mathsf{o}_1 \cdot \text{contextOpen}(1) \cdot \mathsf{c}_1)^*\Big) \ .
\end{aligned}
$$

$$
\begin{aligned}
& \mathsf{o}_0\mathsf{c}_0 + \mathsf{o}_1 \cdot \langle \mathsf{f} \cdot (\!|_{\mathsf{y}_1}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle \cdot \mathsf{c}_1 \\
= \quad & \{\!\{ \text{ Identity of } \cdot \ \}\!\} \\
& \mathsf{o}_0 \cdot 1 \cdot \mathsf{c}_0 + \mathsf{o}_1 \cdot \langle \mathsf{f} \cdot (\!|_{\mathsf{y}_1}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle \cdot \mathsf{c}_1 \\
\leqslant \quad & \{\!\{ \text{ Kleene algebra: for any expression } r,\ 1 \leqslant r^* \ \&\ \text{Monotonicity of } \cdot \text{ and } + \ \}\!\} \\
& \mathsf{o}_0 \cdot \Big(\langle \mathsf{f} \cdot (\!|_{\mathsf{y}_0}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle + \mathsf{o}_1 \cdot (\langle \mathsf{f} \cdot (\!|_{\mathsf{y}_{0,1}}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle)^* \cdot \mathsf{c}_1\Big)^* \cdot \mathsf{c}_0 + \mathsf{o}_1 \cdot \langle \mathsf{f} \cdot (\!|_{\mathsf{y}_1}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle \cdot \mathsf{c}_1 \\
\leqslant \quad & \{\!\{ \text{ Kleene algebra: for any expressions } r_1 \text{ and } r_2,\ r_1 \leqslant (r_1 + r_2)^* \ \&\ \text{Mono-} \\
& \qquad \text{tonicity of } \cdot \text{ and } + \ \}\!\} \\
& \mathsf{o}_0 \cdot \Big(\langle \mathsf{f} \cdot (\!|_{\mathsf{y}_0}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle + \mathsf{o}_1 \cdot (\langle \mathsf{f} \cdot (\!|_{\mathsf{y}_{0,1}}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle)^* \cdot \mathsf{c}_1\Big)^* \cdot \mathsf{c}_0 \\
& + \mathsf{o}_1 \cdot \Big(\langle \mathsf{f} \cdot (\!|_{\mathsf{y}_1}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle + \mathsf{o}_0 \cdot (\langle \mathsf{f} \cdot (\!|_{\mathsf{y}_{0,1}}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle)^* \cdot \mathsf{c}_0\Big)^* \cdot \mathsf{c}_1 \\
\leqslant \quad & \{\!\{ \text{ Kleene algebra: for any expressions } r_1 \text{ and } r_2,\ r_1 \leqslant (r_2 + r_1)^* \ \}\!\} \\
& \Big(\langle \mathsf{f} \cdot (\!|_{\mathsf{x}}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle + \mathsf{o}_0 \cdot \Big(\langle \mathsf{f} \cdot (\!|_{\mathsf{y}_0}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle + \mathsf{o}_1 \cdot (\langle \mathsf{f} \cdot (\!|_{\mathsf{y}_{0,1}}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle)^* \cdot \mathsf{c}_1\Big)^* \cdot \mathsf{c}_0 \\
& \quad + \ \mathsf{o}_1 \cdot \Big(\langle \mathsf{f} \cdot (\!|_{\mathsf{y}_1}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle + \mathsf{o}_0 \cdot (\langle \mathsf{f} \cdot (\!|_{\mathsf{y}_{0,1}}\, \mathcal{C}\, |\!)^{\tau} \cdot \mathsf{f} \rangle)^* \cdot \mathsf{c}_0\Big)^* \cdot \mathsf{c}_1\Big)^*
\end{aligned}
$$

$$
\begin{aligned}
&= && \{\!\{ \text{ Definition of contextOpen}(j) \text{ for } j \in 0,1 \text{ and } \mathsf{mb\_vpre}_{\theta_{\mathcal{C}}} \}\!\} \\
&\quad \mathsf{mb\_vpre}_{\theta_{\mathcal{C}}}\Big((\langle \mathsf{f} \mathsf{x}^{\mathsf{f}} \rangle + \mathsf{o}_0 \cdot \mathrm{contextOpen}(0) \cdot \mathsf{c}_0 + \mathsf{o}_1 \cdot \mathrm{contextOpen}(1) \cdot \mathsf{c}_1)^*\Big)
\end{aligned}
$$

Then, we prove that

$$
\langle \mathsf{f} \cdot (\!|\, \underset{\mathsf{x}}{\mathcal{C}} \,|\!)^{\tau} \cdot \mathsf{f} \rangle \leqslant (\!|\, \underset{\mathsf{x}}{\mathcal{C}} \,|\!)^{\tau} \quad . \tag{6.4}
$$

By inequation (5.1) and the transitivity of $\leqslant$, it suffices to prove that

$$
\begin{aligned}
&\langle \mathsf{f} \cdot (\!|_{\mathsf{x}}\, \mathcal{C} \,|\!)^{\tau} \cdot \mathsf{f} \rangle \\
&\leqslant \ \mathsf{mb\_vpre}_{\theta_{\mathcal{C}}}\Big((\langle \mathsf{f} \mathsf{x}^{\mathsf{f}} \rangle + \mathsf{o}_0 \cdot \mathrm{contextOpen}(0) \cdot \mathsf{c}_0 + \mathsf{o}_1 \cdot \mathrm{contextOpen}(1) \cdot \mathsf{c}_1)^*\Big) \quad .
\end{aligned}
$$

$$
\begin{aligned}
&\langle \mathsf{f} \cdot (\!|_{\mathsf{x}}\, \mathcal{C} \,|\!)^{\tau} \cdot \mathsf{f} \rangle \\
&\leqslant && \{\!\{ \text{ Kleene algebra: for any expressions } r_1 \text{ and } r_2,\ r_1 \leqslant (r_1 + r_2)^* \}\!\} \\
&\quad \Big(\langle \mathsf{f} \cdot (\!|_{\mathsf{x}}\, \mathcal{C} \,|\!)^{\tau} \cdot \mathsf{f} \rangle + \mathsf{o}_0 \cdot \Big(\langle \mathsf{f} \cdot (\!|_{\mathsf{y}_0}\, \mathcal{C} \,|\!)^{\tau} \cdot \mathsf{f} \rangle + \mathsf{o}_1 \cdot (\langle \mathsf{f} \cdot (\!|_{\mathsf{y}_{0,1}}\, \mathcal{C} \,|\!)^{\tau} \cdot \mathsf{f} \rangle)^* \cdot \mathsf{c}_1\Big)^* \cdot \mathsf{c}_0 \\
&\qquad + \ \mathsf{o}_1 \cdot \Big(\langle \mathsf{f} \cdot (\!|_{\mathsf{y}_1}\, \mathcal{C} \,|\!)^{\tau} \cdot \mathsf{f} \rangle + \mathsf{o}_0 \cdot (\langle \mathsf{f} \cdot (\!|_{\mathsf{y}_{0,1}}\, \mathcal{C} \,|\!)^{\tau} \cdot \mathsf{f} \rangle)^* \cdot \mathsf{c}_0\Big)^* \cdot \mathsf{c}_1\Big)^* \\
&= && \{\!\{ \text{ Definition of contextOpen}(j) \text{ for } j \in 0,1 \text{ and } \mathsf{mb\_vpre}_{\theta_{\mathcal{C}}} \}\!\} \\
&\quad \mathsf{mb\_vpre}_{\theta_{\mathcal{C}}}\Big((\langle \mathsf{f} \mathsf{x}^{\mathsf{f}} \rangle + \mathsf{o}_0 \cdot \mathrm{contextOpen}(0) \cdot \mathsf{c}_0 + \mathsf{o}_1 \cdot \mathrm{contextOpen}(1) \cdot \mathsf{c}_1)^*\Big)
\end{aligned}
$$

We are now ready to prove that $\mathcal{H} \wedge \mathcal{R} \to p \leqslant q$.

$$
\begin{aligned}
&(\!|_{\mathsf{f}'}\, \mathcal{B} \,|\!)^{\tau} \\
&= && \{\!\{ \text{ Equations (5.24) and (6.1) \& Zero of } \cdot \text{ \& Identity of } + \}\!\} \\
&\quad \langle \mathsf{f} \cdot (\!|_{\mathsf{f}}\, \mathcal{B} \,|\!)^{\tau} \cdot \mathsf{f} \rangle \\
&= && \{\!\{ \text{ Equations (5.24) and (6.1) \& Zero of } \cdot \text{ \& Identity of } + \}\!\} \\
&\quad \langle \mathsf{f} \cdot (\mathsf{boc} + \overline{\mathsf{b}}\mathsf{o} \cdot \langle \mathsf{f} \cdot (\!|_{\mathsf{f}}\, \mathcal{B} \,|\!)^{\tau} \cdot \mathsf{f} \rangle \cdot \mathsf{c}) \cdot \mathsf{f} \rangle \\
&\leqslant && \{\!\{ \text{ Idempotency of tests \& Hypotheses in } \mathcal{H} \text{ \& Kleene algebra with tests:} \\
&\qquad \mathsf{bo} = \mathsf{ob} \leftrightarrow \overline{\mathsf{b}}\mathsf{o} = \mathsf{o}\overline{\mathsf{b}} \text{ \& Monotonicity of } \cdot \text{ and } + \}\!\} \\
&\quad \langle \mathsf{f} \cdot (\mathsf{bobc} + \overline{\mathsf{b}}\mathsf{o} \cdot \langle \mathsf{f} \cdot \mathsf{b} \cdot (\!|_{\mathsf{f}}\, \mathcal{B} \,|\!)^{\tau} \cdot \mathsf{f} \rangle \cdot \overline{\mathsf{b}}\mathsf{c}) \cdot \mathsf{f} \rangle \\
&= && \{\!\{ \text{ Equations (5.24) and (6.1) \& Zero of } \cdot \text{ \& Identity of } + \}\!\} \\
&\quad \langle \mathsf{f} \cdot \Big(\mathsf{bobc} + \overline{\mathsf{b}}\mathsf{o} \cdot \langle \mathsf{f} \cdot \mathsf{b} \cdot (\mathsf{boc} + \overline{\mathsf{b}}\mathsf{o} \cdot \langle \mathsf{f} \cdot (\!|_{\mathsf{f}}\, \mathcal{B} \,|\!)^{\tau} \cdot \mathsf{f} \rangle \cdot \mathsf{c}) \cdot \mathsf{f} \rangle \cdot \overline{\mathsf{b}}\mathsf{c}\Big) \cdot \mathsf{f} \rangle \\
&= && \{\!\{ \text{ Distributivity of } \cdot \text{ over } + \text{ \& Contradiction of tests \& Zero of } \cdot \text{ \& Identity} \\
&\qquad \text{of } + \text{ \& Hypothesis: } \mathsf{bo} = \mathsf{ob} \}\!\}
\end{aligned}
$$

$$
\begin{aligned}
& \langle \mathsf{f} \cdot (\mathsf{bobc} + \overline{\mathsf{b}}\mathsf{o} \cdot \langle \mathsf{f} \cdot \mathsf{bobc} \cdot \mathsf{f} \rangle \cdot \overline{\mathsf{b}}\mathsf{c}) \cdot \mathsf{f} \rangle \\
\leqslant & \qquad \{\!\{ \text{Hypotheses in } \mathcal{R} \text{ \& Monotonicity of } \cdot \text{ and } + \}\!\} \\
& \langle \mathsf{f} \cdot (\mathsf{o}_0\mathsf{c}_0 + \mathsf{o}_1 \cdot \langle \mathsf{f} \cdot \mathsf{o}_0\mathsf{c}_0 \cdot \mathsf{f} \rangle \cdot \mathsf{c}_1) \cdot \mathsf{f} \rangle \\
\leqslant & \qquad \{\!\{ \text{Inequation (6.2) \& Monotonicity of } \cdot \text{ and } + \}\!\} \\
& \langle \mathsf{f} \cdot (\mathsf{o}_0\mathsf{c}_0 + \mathsf{o}_1 \cdot \langle \mathsf{f} \cdot (\!|_{\mathsf{y}_1} \mathcal{C} |\!)^{\tau} \cdot \mathsf{f} \rangle \cdot \mathsf{c}_1) \cdot \mathsf{f} \rangle \\
\leqslant & \qquad \{\!\{ \text{Inequation (6.3) \& Monotonicity of } \cdot \}\!\} \\
& \langle \mathsf{f} \cdot (\!|_{\mathsf{x}} \mathcal{C} |\!)^{\tau} \cdot \mathsf{f} \rangle \\
\leqslant & \qquad \{\!\{ \text{Inequation (6.4)} \}\!\} \\
& (\!|_{\mathsf{x}} \mathcal{C} |\!)^{\tau}
\end{aligned}
$$

## 6.3 Verification of Common Interprocedural Compiler Optimizations

Let us examine how some interprocedural compiler optimizations can be verified with VPKAT.[1] Our proposition for this verification (which is an extension of Kozen and Patron's proposition [33]) is shown in Figure 6.8. In this proposition, the encoding of the unoptimized and optimized programs in VPKAT follows the encoding presented in Section 6.1. Note that the set of hypotheses $\mathcal{H}$ contains not only the desired semantics of atomic program instructions, variables and variable passing mechanisms, but may also contain some *optimization assumptions* (assumptions about the optimizing transformations that have been applied). Optimization assumptions will be used in Sections 6.3.2 and 6.3.3.

The verification of a sequence of interprocedural optimizing transformations is done by verifying if $\mathcal{H} \to p = q$ is a theorem of VPKAT. The optimization is considered safe if and only if the formula is a theorem of VPKAT (after all, the formula expresses that the two programs are "semantically" equivalent under hypotheses $\mathcal{H}$). Our verification deals only with halting programs, since non-halting programs may lead to incorrect results. Halting programs are obtained by restricting recursive procedures and loops to simple cases.

We now present some interprocedural compiler optimizations and we show how they can be verified using VPKAT.

[1] We do not verify the compiler itself, but we verify the *output* of a compiler after it has done some optimizations.

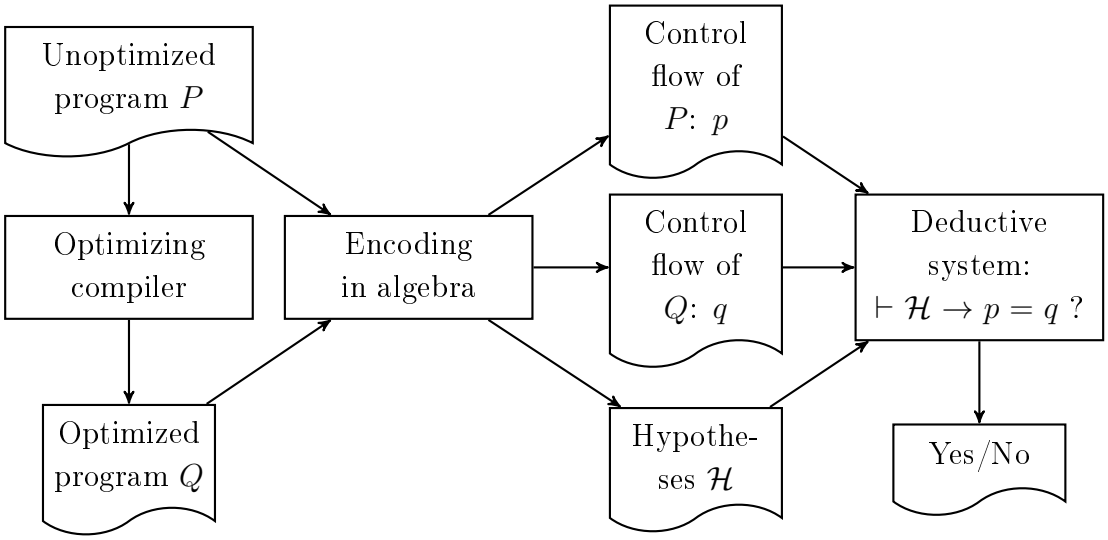



Figure 6.8: Verification of interprocedural compiler optimizations in visibly pushdown Kleene algebra with tests.

### 6.3.1 Interprocedural Dead Code Elimination

Dead code elimination is the removal of unreachable instructions. When dealing with procedures, it is possible that all actual calls of a procedure in a program do not allow it to reach a set of instructions. There is a common case when this situation happens: the case where the preconditions of a procedure are not respected.

#### Simple Non Recursive Example

Take the C program of Figure 6.9a. In this program, the pointer n cannot be null. So, the *compiler* can remove the test n == NULL without modifying the behaviour of the code. This will speed up the application (not too much in this example, but it can be very handy in real situations). However, note that it is useful for a *programmer* to write this test to have a bullet-proof code and ease maintenance over time. So, it is not the task of the programmer to remove this test, but the task of the compiler!

Figure 6.9b is the C program without the test[2]. Using visibly pushdown Kleene algebra, it is possible to prove that the optimized program is equivalent to the unoptimized one. Let b be the representation of the test n != NULL. Also, let a represent the test

[2]Of course, in this code the remaining increment function should be inlined by the compiler to make it more efficient. We will talk about this issue in Section 6.3.2.

```
int main(void) {
  int x = 2;

  increment(&x);
  increment(&x);

  return 0; /*Success. */
}

void increment(int* n) {
  if(n == NULL)
    printf(stderr, "Error");
  else
    *n += 1;
}
```

(a) Unoptimized program

```
int main(void) {
  int x = 2;

  increment(&x);
  increment(&x);

  return 0; /*Success. */
}

void increment(int* n) {



  *n += 1;
}
```

(b) Optimized program

Figure 6.9: Simple non-recursive program example.

that the address of the variable x exists. Let

- p be the internal action printf(stderr, "Error");
- q be the internal action $*$n += 1;
- s be the internal action **int** x = 2;
- t be the internal action **return** 0. Of course, it is a return action in the code, but we treat it as an internal action *followed by* a return action (going out of context). This will be clear in the encoding of the program control flow.

Since s creates the local variable x, we have the hypothesis $\mathsf{s} = \mathsf{s} \cdot \mathsf{a}$. We also use the hypothesis $\mathsf{a} \cdot \mathsf{t} = \mathsf{t}$ since the memory for x is still allocated just before the return of the context (it is a local variable). Moreover, the action $*$n += 1 modifies neither the test n != NULL nor the existence of x. So, the hypothesis $\mathsf{a} \cdot \mathsf{b} \cdot \mathsf{q} = \mathsf{q} \cdot \mathsf{a} \cdot \mathsf{b}$ is correct in this program.

For the representation of the function calls, let $\langle$m and m$\rangle$ be respectively the call of the main function and its return. Also, let $\langle$f and f$\rangle$ be respectively the call of the increment function and its return.

Since the variable $\mathsf{x}$ is passed to the function $\mathsf{increment}$ by a pointer, we have the following valid hypotheses: $\mathsf{a} \cdot \langle \mathsf{f} = \langle \mathsf{f} \cdot \mathsf{a} \cdot \mathsf{b}$ and $\mathsf{a} \cdot \mathsf{b} \cdot \mathsf{f}\rangle = \mathsf{f}\rangle \cdot \mathsf{a}$.

With these actions and hypotheses, the control flow of the two programs can easily be encoded in visibly pushdown Kleene algebra. The first program gives

$$\Big(\!\Big|\, \underset{\mathsf{m}'}{\big[}\, \underset{\mathsf{m}'}{\langle \mathsf{m} \rangle}\, \underset{\mathsf{m}}{\big]^\tau}, \underset{\mathsf{m}}{\big[}\, \mathsf{s} \cdot \langle \mathsf{f} \rangle \cdot \langle \mathsf{f} \rangle \cdot \mathsf{t}\, \underset{\mathsf{f}}{\big]^\tau}, \underset{\mathsf{f}}{\big[}\, \overline{\mathsf{b}} \cdot \mathsf{p} + \mathsf{b} \cdot \mathsf{q}\, \big]^{\tau\tau} \,\Big|\!\Big) \ .$$

So, we must show that, under the preceding hypotheses,

$$\Big(\!\Big|\, \underset{\mathsf{m}'}{\big[}\, \underset{\mathsf{m}'}{\langle \mathsf{m} \rangle}\, \underset{\mathsf{m}}{\big]^\tau}, \underset{\mathsf{m}}{\big[}\, \mathsf{s} \cdot \langle \mathsf{f} \rangle \cdot \langle \mathsf{f} \rangle \cdot \mathsf{t}\, \underset{\mathsf{f}}{\big]^\tau}, \underset{\mathsf{f}}{\big[}\, \overline{\mathsf{b}} \cdot \mathsf{p} + \mathsf{b} \cdot \mathsf{q}\, \big]^{\tau\tau} \,\Big|\!\Big) = \Big(\!\Big|\, \underset{\mathsf{m}'}{\big[}\, \underset{\mathsf{m}'}{\langle \mathsf{m} \rangle}\, \underset{\mathsf{m}}{\big]^\tau}, \underset{\mathsf{m}}{\big[}\, \mathsf{s} \cdot \langle \mathsf{f} \rangle \cdot \langle \mathsf{f} \rangle \cdot \mathsf{t}\, \underset{\mathsf{f}}{\big]^\tau}, \underset{\mathsf{f}}{\big[}\, \mathsf{q}\, \big]^{\tau\tau} \,\Big|\!\Big) \ . \tag{6.5}$$

First, note that no valid derivation can *start* with $\tau$. So, by (5.24),

$$\Big(\!\Big|\, \underset{\tau\,\mathsf{m}'}{\big[}\, \underset{\mathsf{m}'}{\langle \mathsf{m} \rangle}\, \underset{\mathsf{m}}{\big]^\tau}, \underset{\mathsf{m}}{\big[}\, \mathsf{s} \cdot \langle \mathsf{f} \rangle \cdot \langle \mathsf{f} \rangle \cdot \mathsf{t}\, \underset{\mathsf{f}}{\big]^\tau}, \underset{\mathsf{f}}{\big[}\, \overline{\mathsf{b}} \cdot \mathsf{p} + \mathsf{b} \cdot \mathsf{q}\, \big]^{\tau\tau} \,\Big|\!\Big) = 0 \ . \tag{6.6}$$

Now, let us prove (6.5).

$$\begin{aligned}
& \big(\!\big|_{\mathsf{m}'} \big[_{\mathsf{m}'} \langle \mathsf{m} \rangle \big]^\tau, \big[_{\mathsf{m}} \mathsf{s} \cdot \langle \mathsf{f} \rangle \cdot \langle \mathsf{f} \rangle \cdot \mathsf{t} \big]^\tau, \big[_{\mathsf{f}} \overline{\mathsf{b}} \cdot \mathsf{p} + \mathsf{b} \cdot \mathsf{q} \big]^\tau \big|\!\big)^\tau \\
= \quad & \{\!\{ \text{ Equations (5.24) and (6.6) } \}\!\} \\
& \langle \mathsf{m} \cdot \big(\!\big|_{\mathsf{m}} \big[_{\mathsf{m}'} \langle \mathsf{m} \rangle \big]^\tau, \big[_{\mathsf{m}} \mathsf{s} \cdot \langle \mathsf{f} \rangle \cdot \langle \mathsf{f} \rangle \cdot \mathsf{t} \big]^\tau, \big[_{\mathsf{f}} \overline{\mathsf{b}} \cdot \mathsf{p} + \mathsf{b} \cdot \mathsf{q} \big]^\tau \big|\!\big)^\tau \cdot \mathsf{m} \rangle \\
= \quad & \{\!\{ \text{ Equations (5.24) and (6.6) } \}\!\} \\
& \langle \mathsf{m} \cdot \mathsf{s} \cdot \langle \mathsf{f} \cdot \big(\!\big|_{\mathsf{f}} \big[_{\mathsf{m}'} \langle \mathsf{m} \rangle \big]^\tau, \big[_{\mathsf{m}} \mathsf{s} \cdot \langle \mathsf{f} \rangle \cdot \langle \mathsf{f} \rangle \cdot \mathsf{t} \big]^\tau, \big[_{\mathsf{f}} \overline{\mathsf{b}} \cdot \mathsf{p} + \mathsf{b} \cdot \mathsf{q} \big]^\tau \big|\!\big)^\tau \cdot \mathsf{f} \rangle \\
& \quad \cdot \langle \mathsf{f} \cdot \big(\!\big|_{\mathsf{f}} \big[_{\mathsf{m}'} \langle \mathsf{m} \rangle \big]^\tau, \big[_{\mathsf{m}} \mathsf{s} \cdot \langle \mathsf{f} \rangle \cdot \langle \mathsf{f} \rangle \cdot \mathsf{t} \big]^\tau, \big[_{\mathsf{f}} \overline{\mathsf{b}} \cdot \mathsf{p} + \mathsf{b} \cdot \mathsf{q} \big]^\tau \big|\!\big)^\tau \cdot \mathsf{f} \rangle \cdot \mathsf{t} \cdot \mathsf{m} \rangle \\
= \quad & \{\!\{ \text{ Equations (5.24) and (6.6), multiple times } \}\!\} \\
& \langle \mathsf{m} \cdot \mathsf{s} \cdot \langle \mathsf{f} \cdot (\overline{\mathsf{b}} \cdot \mathsf{p} + \mathsf{b} \cdot \mathsf{q}) \cdot \mathsf{f} \rangle \cdot \langle \mathsf{f} \cdot (\overline{\mathsf{b}} \cdot \mathsf{p} + \mathsf{b} \cdot \mathsf{q}) \cdot \mathsf{f} \rangle \cdot \mathsf{t} \cdot \mathsf{m} \rangle \\
= \quad & \{\!\{ \text{ Hypothesis: } \mathsf{s} = \mathsf{s} \cdot \mathsf{a} \ \}\!\} \\
& \langle \mathsf{m} \cdot \mathsf{s} \cdot \mathsf{a} \cdot \langle \mathsf{f} \cdot (\overline{\mathsf{b}} \cdot \mathsf{p} + \mathsf{b} \cdot \mathsf{q}) \cdot \mathsf{f} \rangle \cdot \langle \mathsf{f} \cdot (\overline{\mathsf{b}} \cdot \mathsf{p} + \mathsf{b} \cdot \mathsf{q}) \cdot \mathsf{f} \rangle \cdot \mathsf{t} \cdot \mathsf{m} \rangle \\
= \quad & \{\!\{ \text{ Hypothesis: } \mathsf{a} \cdot \langle \mathsf{f} = \langle \mathsf{f} \cdot \mathsf{a} \cdot \mathsf{b} \ \}\!\} \\
& \langle \mathsf{m} \cdot \mathsf{s} \cdot \langle \mathsf{f} \cdot \mathsf{a} \cdot \mathsf{b} \cdot (\overline{\mathsf{b}} \cdot \mathsf{p} + \mathsf{b} \cdot \mathsf{q}) \cdot \mathsf{f} \rangle \cdot \langle \mathsf{f} \cdot (\overline{\mathsf{b}} \cdot \mathsf{p} + \mathsf{b} \cdot \mathsf{q}) \cdot \mathsf{f} \rangle \cdot \mathsf{t} \cdot \mathsf{m} \rangle \\
= \quad & \{\!\{ \text{ Distributivity of } \cdot \text{ over } + \ \}\!\} \\
& \langle \mathsf{m} \cdot \mathsf{s} \cdot \langle \mathsf{f} \cdot (\mathsf{a} \cdot \mathsf{b} \cdot \overline{\mathsf{b}} \cdot \mathsf{p} + \mathsf{a} \cdot \mathsf{b} \cdot \mathsf{b} \cdot \mathsf{q}) \cdot \mathsf{f} \rangle \cdot \langle \mathsf{f} \cdot (\overline{\mathsf{b}} \cdot \mathsf{p} + \mathsf{b} \cdot \mathsf{q}) \cdot \mathsf{f} \rangle \cdot \mathsf{t} \cdot \mathsf{m} \rangle \\
= \quad & \{\!\{ \text{ Contradiction of tests \& Idempotency of tests } \}\!\} \\
& \langle \mathsf{m} \cdot \mathsf{s} \cdot \langle \mathsf{f} \cdot (\mathsf{a} \cdot 0 \cdot \mathsf{p} + \mathsf{a} \cdot \mathsf{b} \cdot \mathsf{q}) \cdot \mathsf{f} \rangle \cdot \langle \mathsf{f} \cdot (\overline{\mathsf{b}} \cdot \mathsf{p} + \mathsf{b} \cdot \mathsf{q}) \cdot \mathsf{f} \rangle \cdot \mathsf{t} \cdot \mathsf{m} \rangle \\
= \quad & \{\!\{ \text{ Zero of } \cdot \text{ \& Identity of } + \ \}\!\} \\
& \langle \mathsf{m} \cdot \mathsf{s} \cdot \langle \mathsf{f} \cdot \mathsf{a} \cdot \mathsf{b} \cdot \mathsf{q} \cdot \mathsf{f} \rangle \cdot \langle \mathsf{f} \cdot (\overline{\mathsf{b}} \cdot \mathsf{p} + \mathsf{b} \cdot \mathsf{q}) \cdot \mathsf{f} \rangle \cdot \mathsf{t} \cdot \mathsf{m} \rangle \\
= \quad & \{\!\{ \text{ Hypotheses: } \mathsf{a} \cdot \mathsf{b} \cdot \mathsf{q} = \mathsf{q} \cdot \mathsf{a} \cdot \mathsf{b}, \mathsf{a} \cdot \mathsf{b} \cdot \mathsf{f}\rangle = \mathsf{f}\rangle \cdot \mathsf{a} \text{ and } \mathsf{a} \cdot \langle \mathsf{f} = \langle \mathsf{f} \cdot \mathsf{a} \cdot \mathsf{b} \ \}\!\} \\
& \langle \mathsf{m} \cdot \mathsf{s} \cdot \langle \mathsf{f} \cdot \mathsf{q} \cdot \mathsf{f} \rangle \cdot \langle \mathsf{f} \cdot \mathsf{a} \cdot \mathsf{b} \cdot (\overline{\mathsf{b}} \cdot \mathsf{p} + \mathsf{b} \cdot \mathsf{q}) \cdot \mathsf{f} \rangle \cdot \mathsf{t} \cdot \mathsf{m} \rangle \\
= \quad & \{\!\{ \text{ Distributivity of } \cdot \text{ over } + \text{ \& Contradiction of tests \& Idempotency of} \\
& \quad \text{tests } \}\!\}
\end{aligned}$$

$$
\begin{aligned}
& \langle \mathsf{m} \cdot \mathsf{s} \cdot \langle \mathsf{f} \cdot \mathsf{q} \cdot \mathsf{f} \rangle \cdot \langle \mathsf{f} \cdot (\mathsf{a} \cdot 0 \cdot \mathsf{p} + \mathsf{a} \cdot \mathsf{b} \cdot \mathsf{q}) \cdot \mathsf{f} \rangle \cdot \mathsf{t} \cdot \mathsf{m} \rangle \\
= & \qquad \{\!\{ \text{ Zero of } \cdot \text{ \& Identity of } + \ \}\!\} \\
& \langle \mathsf{m} \cdot \mathsf{s} \cdot \langle \mathsf{f} \cdot \mathsf{q} \cdot \mathsf{f} \rangle \cdot \langle \mathsf{f} \cdot \mathsf{a} \cdot \mathsf{b} \cdot \mathsf{q} \cdot \mathsf{f} \rangle \cdot \mathsf{t} \cdot \mathsf{m} \rangle \\
= & \qquad \{\!\{ \text{ Hypotheses: } \mathsf{a} \cdot \mathsf{b} \cdot \mathsf{q} = \mathsf{q} \cdot \mathsf{a} \cdot \mathsf{b},\ \mathsf{a} \cdot \mathsf{b} \cdot \mathsf{f} \rangle = \mathsf{f} \rangle \cdot \mathsf{a} \text{ and } \mathsf{a} \cdot \mathsf{t} = \mathsf{t} \ \}\!\} \\
& \langle \mathsf{m} \cdot \mathsf{s} \cdot \langle \mathsf{f} \cdot \mathsf{q} \cdot \mathsf{f} \rangle \cdot \langle \mathsf{f} \cdot \mathsf{q} \cdot \mathsf{f} \rangle \cdot \mathsf{t} \cdot \mathsf{m} \rangle \\
= & \qquad \{\!\{ \text{ Equations (5.24) and (6.6) } \}\!\} \\
& \langle \mathsf{m} \cdot \mathsf{s} \cdot \langle \mathsf{f} \cdot (\!|_{\mathsf{f}} [_{\mathsf{m}'} \langle \mathsf{m} \rangle ]^{\tau}, [_{\mathsf{m}} \mathsf{s} \cdot \langle \mathsf{f} \rangle \cdot \langle \mathsf{f} \rangle \cdot \mathsf{t} ]^{\tau}, [_{\mathsf{f}} \mathsf{q} ]^{\tau} |)^{\tau} \cdot \mathsf{f} \rangle \\
& \qquad \cdot \langle \mathsf{f} \cdot (\!|_{\mathsf{f}} [_{\mathsf{m}'} \langle \mathsf{m} \rangle ]^{\tau}, [_{\mathsf{m}} \mathsf{s} \cdot \langle \mathsf{f} \rangle \cdot \langle \mathsf{f} \rangle \cdot \mathsf{t} ]^{\tau}, [_{\mathsf{f}} \mathsf{q} ]^{\tau} |)^{\tau} \cdot \mathsf{f} \rangle \cdot \mathsf{t} \cdot \mathsf{m} \rangle \\
= & \qquad \{\!\{ \text{ Equations (5.24) and (6.6) } \}\!\} \\
& \langle \mathsf{m} \cdot (\!|_{\mathsf{m}} [_{\mathsf{m}'} \langle \mathsf{m} \rangle ]^{\tau}, [_{\mathsf{m}} \mathsf{s} \cdot \langle \mathsf{f} \rangle \cdot \langle \mathsf{f} \rangle \cdot \mathsf{t} ]^{\tau}, [_{\mathsf{f}} \mathsf{q} ]^{\tau} |)^{\tau} \cdot \mathsf{m} \rangle \\
= & \qquad \{\!\{ \text{ Equations (5.24) and (6.6) } \}\!\} \\
& (\!|_{\mathsf{m}'} [_{\mathsf{m}'} \langle \mathsf{m} \rangle ]^{\tau}, [_{\mathsf{m}} \mathsf{s} \cdot \langle \mathsf{f} \rangle \cdot \langle \mathsf{f} \rangle \cdot \mathsf{t} ]^{\tau}, [_{\mathsf{f}} \mathsf{q} ]^{\tau} |)^{\tau}
\end{aligned}
$$

### Complex Recursive Example

The verification of interprocedural dead code elimination works also in the presence of recursive functions and user inputs when considering only halting cases. For example, take the C program of Figure 6.10a that calculates the factorial of a number. In that program, the test $\mathsf{n} < 0$ in the function fact cannot be true. So, it can be removed as in Figure 6.10b.

We can prove that these two programs are equivalent using visibly pushdown Kleene algebra with tests. If we are only interested in a bounded value for n, then the proof follows a pattern similar to the proof of page 139. However, the unbounded case for n is also provable. Here is such a proof.

By abstract interpretation, it is easy to see that it is possible to limit ourselves to the equivalence classes

$$[-\infty, -1], \quad \{0\}, \quad \{1\}, \quad [2, +\infty]$$

for x in this verification. Note that this abstraction does not allow us to calculate the real answer of the factorial, but we are not interested in this. We are just interested in removing the test $\mathsf{n} < 0$.

Let us encode variable x by the atomic tests $a_0$ and $b_0$ (since there are four cases to represent for x) and variable n by the atomic tests $a$ and $b$ (since there is four cases to represent for n):

```
int main(void) {
  int x, result ;

  do {
    /* Ask a number. */
    scanf("%d", &x);
  } while(x < 0)

  result = fact(x);
  printf ("%d", result );

  return 0; /*Success. */
}

int fact(int n) {
  int prec_fact;
  if(n < 0)
    return −1; /*Error. */
  else {
    if(n == 0)
      return 1;
    else {
      prec_fact = fact(n−1);
      return n ∗ prec_fact;
    }
  }
}
```

(a) Unoptimized program

```
int main(void) {
  int x, result ;

  do {
    /* Ask a number. */
    scanf("%d", &x);
  } while(x < 0)

  result = fact(x);
  printf ("%d", result );

  return 0; /*Success. */
}

int fact(int n) {
  int prec_fact;




  if(n == 0)
    return 1;
  else {
    prec_fact = fact(n−1);
    return n ∗ prec_fact;
  }

}
```

(b) Optimized program

Figure 6.10: Complex recursive program example.

- $\overline{a_0}\overline{b_0}$ represents the case $x < 0$;
- $\overline{a_0}b_0$ represents the case $x == 0$;
- $a_0\overline{b_0}$ represents the case $x == 1$;
- $a_0b_0$ represents the case $x > 1$;
- $\overline{a}\overline{b}$ represents the case $n < 0$;

- $\bar{\mathsf{a}}\mathsf{b}$ represents the case $\mathsf{n} == 0$;
- $\mathsf{a}\bar{\mathsf{b}}$ represents the case $\mathsf{n} == 1$;
- $\mathsf{ab}$ represents the case $\mathsf{n} > 1$.

Also, let

- $\mathsf{p}$ be the internal action $\mathsf{scanf("\%d", \&x)}$;
- $\mathsf{q}$ be the internal action $\mathsf{result} = \mathsf{fact(x)}$ (here, it is the assignment that is encoded and not the call to the $\mathsf{fact}$ function);
- $\mathsf{r}$ be the internal action $\mathsf{printf("\%d", result)}$;
- $\mathsf{s}$ be the internal action **return** $0$;
- $\mathsf{t}$ be the internal action **return** $-1$;
- $\mathsf{u}$ be the internal action **return** $1$;
- $\mathsf{v}$ be the internal action $\mathsf{prec\_fact} = \mathsf{fact(n - 1)}$;
- $\mathsf{w}$ be the internal action **return** $(\mathsf{n} * \mathsf{prec\_fact})$.

We also need some hypotheses on the representation of the function calls. Let $\langle \mathsf{f}_1$ and $\mathsf{f}_1 \rangle$ be respectively the call and return of the $\mathsf{fact}$ function by the $\mathsf{main}$ function. Let $\langle \mathsf{f}_2$ and $\mathsf{f}_2 \rangle$ be respectively the call and return of the $\mathsf{fact}$ function by the $\mathsf{fact}$ function. We distinguish between the call of the $\mathsf{main}$ function and the call of the $\mathsf{fact}$ function because they do not have the same argument (one is $\mathsf{n}$ and the other is $\mathsf{n} - 1$).

We have the following valid hypotheses for the call actions:

$$\begin{aligned}
&\overline{\mathsf{a}_0}\overline{\mathsf{b}_0} \cdot \langle \mathsf{f}_1 = \langle \mathsf{f}_1 \cdot \bar{\mathsf{a}}\bar{\mathsf{b}} \ , \\
&\overline{\mathsf{a}_0}\mathsf{b}_0 \cdot \langle \mathsf{f}_1 = \langle \mathsf{f}_1 \cdot \bar{\mathsf{a}}\mathsf{b} \ , \\
&\mathsf{a}_0\overline{\mathsf{b}_0} \cdot \langle \mathsf{f}_1 = \langle \mathsf{f}_1 \cdot \mathsf{a}\bar{\mathsf{b}} \ , \\
&\mathsf{a}_0\mathsf{b}_0 \cdot \langle \mathsf{f}_1 = \langle \mathsf{f}_1 \cdot \mathsf{ab} \ , \\
&\mathsf{ab} \cdot \langle \mathsf{f}_2 = \langle \mathsf{f}_2 \cdot (\mathsf{ab} + \mathsf{a}\bar{\mathsf{b}}) \ , \\
&\mathsf{a}\bar{\mathsf{b}} \cdot \langle \mathsf{f}_2 = \langle \mathsf{f}_2 \cdot \bar{\mathsf{a}}\mathsf{b} \ , \\
&\bar{\mathsf{a}}\mathsf{b} \cdot \langle \mathsf{f}_2 = \langle \mathsf{f}_2 \cdot \bar{\mathsf{a}}\bar{\mathsf{b}} \ , \\
&\bar{\mathsf{a}}\bar{\mathsf{b}} \cdot \langle \mathsf{f}_2 = \langle \mathsf{f}_2 \cdot \bar{\mathsf{a}}\bar{\mathsf{b}} \ .
\end{aligned}$$

The first four hypotheses encode the fact that the variable $\mathsf{x}$ is passed by value to the function $\mathsf{fact}$. Also, the hypothesis $\mathsf{ab} \cdot \langle \mathsf{f}_2 = \langle \mathsf{f}_2 \cdot (\mathsf{ab} + \mathsf{a}\bar{\mathsf{b}})$ states that if $\mathsf{n}$ has a value

greater than 1 before doing the inductive call $\langle \mathsf{f}_2$ (with argument $\mathsf{n} - 1$), then after the call, the value of $\mathsf{n}$ is a value greater than or equal to 1 (and vice versa). Of course, we could have additional hypotheses, but we will not need them in the verification.

With these actions, the two programs can easily be encoded in visibly pushdown Kleene algebra. We must show that, under the preceding hypotheses,

$$\begin{aligned} &\Big(\!\Big|_{\mathsf{m}'}\Big[_{\mathsf{m}'} \langle \mathsf{m} \rangle \Big]^{\tau}, \Big[_{\mathsf{m}} \mathsf{p} \cdot (\overline{\mathsf{a}_0}\overline{\mathsf{b}_0} \cdot \mathsf{p})^* \cdot \overline{\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}} \cdot \langle_1 \mathsf{f}^1 \rangle \cdot \mathsf{q} \cdot \mathsf{r} \cdot \mathsf{s} \Big]^{\tau}, \\ &\quad \Big[_{\mathsf{f}} \overline{\mathsf{a}\mathsf{b}} \cdot \mathsf{t} + \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot (\mathsf{a}\mathsf{b} \cdot \mathsf{u} + \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot \langle_2 \mathsf{f}^2 \rangle \cdot \mathsf{v} \cdot \mathsf{w}) \Big]^{\tau\tau} \Big|\!\Big) \\ &= \Big(\!\Big|_{\mathsf{m}'}\Big[_{\mathsf{m}'} \langle \mathsf{m} \rangle \Big]^{\tau}, \Big[_{\mathsf{m}} \mathsf{p} \cdot (\overline{\mathsf{a}_0}\overline{\mathsf{b}_0} \cdot \mathsf{p})^* \cdot \overline{\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}} \cdot \langle_1 \mathsf{f}^1 \rangle \cdot \mathsf{q} \cdot \mathsf{r} \cdot \mathsf{s} \Big]^{\tau}, \\ &\quad \Big[_{\mathsf{f}} \overline{\mathsf{a}\mathsf{b}} \cdot \mathsf{u} + \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot \langle_2 \mathsf{f}^2 \rangle \cdot \mathsf{v} \cdot \mathsf{w} \Big]^{\tau\tau} \Big|\!\Big) \quad . \end{aligned} \tag{6.7}$$

We first note that

$$\overline{\overline{\mathsf{a}\mathsf{b}}} \cdot \Big(\!\Big|_{\mathsf{f}\,\mathsf{f}} \Big[ \overline{\mathsf{a}\mathsf{b}} \cdot \mathsf{t} + \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot (\mathsf{a}\mathsf{b} \cdot \mathsf{u} + \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot \langle_2 \mathsf{f}^2 \rangle \cdot \mathsf{v} \cdot \mathsf{w}) \Big]^{\tau\tau} \Big|\!\Big) = \Big(\!\Big|_{\mathsf{f}\,\mathsf{f}} \Big[ \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot (\overline{\mathsf{a}\mathsf{b}} \cdot \mathsf{t} + \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot (\mathsf{a}\mathsf{b} \cdot \mathsf{u} + \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot \langle_2 \mathsf{f}^2 \rangle \cdot \mathsf{v} \cdot \mathsf{w})) \Big]^{\tau\tau} \Big|\!\Big) \quad . \tag{6.8}$$

The proof of (6.8) is shown below in page 144.

We now note that

$$\Big(\!\Big|_{\mathsf{f}\,\mathsf{f}} \Big[ \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot (\overline{\mathsf{a}\mathsf{b}} \cdot \mathsf{t} + \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot (\mathsf{a}\mathsf{b} \cdot \mathsf{u} + \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot \langle_2 \mathsf{f}^2 \rangle \cdot \mathsf{v} \cdot \mathsf{w})) \Big]^{\tau\tau} \Big|\!\Big) = \Big(\!\Big|_{\mathsf{f}\,\mathsf{f}} \Big[ \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot (\overline{\mathsf{a}\mathsf{b}} \cdot \mathsf{u} + \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot \langle_2 \mathsf{f}^2 \rangle \cdot \mathsf{v} \cdot \mathsf{w}) \Big]^{\tau\tau} \Big|\!\Big) \quad . \tag{6.9}$$

The proof of (6.9) is shown below in page 146.

Furthermore,

$$\Big(\!\Big|_{\mathsf{f}\,\mathsf{f}} \Big[ \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot (\overline{\mathsf{a}\mathsf{b}} \cdot \mathsf{u} + \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot \langle_2 \mathsf{f}^2 \rangle \cdot \mathsf{v} \cdot \mathsf{w}) \Big]^{\tau\tau} \Big|\!\Big) = \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot \Big(\!\Big|_{\mathsf{f}\,\mathsf{f}} \Big[ \overline{\mathsf{a}\mathsf{b}} \cdot \mathsf{u} + \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot \langle_2 \mathsf{f}^2 \rangle \cdot \mathsf{v} \cdot \mathsf{w} \Big]^{\tau\tau} \Big|\!\Big) \quad . \tag{6.10}$$

The proof of (6.10) is shown below in page 147.

With these results, we are able to prove (6.7).

$$\begin{aligned} &\Big(\!\Big|_{\mathsf{m}'} \Big[_{\mathsf{m}'} \langle \mathsf{m} \rangle \Big]^{\tau}, \Big[_{\mathsf{m}} \mathsf{p} \cdot (\overline{\mathsf{a}_0}\overline{\mathsf{b}_0} \cdot \mathsf{p})^* \cdot \overline{\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}} \cdot \langle_1 \mathsf{f}^1 \rangle \cdot \mathsf{q} \cdot \mathsf{r} \cdot \mathsf{s} \Big]^{\tau}, \Big[_{\mathsf{f}} \overline{\mathsf{a}\mathsf{b}} \cdot \mathsf{t} + \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot (\overline{\mathsf{a}\mathsf{b}} \cdot \mathsf{u} + \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot \langle_2 \mathsf{f}^2 \rangle \cdot \mathsf{v} \cdot \mathsf{w}) \Big]^{\tau} \Big|\!\Big)^{\tau} \\ = \quad & \{\!\{ \text{ Equation (5.24) \& Remark: no valid derivation can } \textit{start} \text{ with } \tau \ \}\!\} \\ & \langle \mathsf{m} \cdot \Big(\!\Big|_{\mathsf{m}} \Big[_{\mathsf{m}'} \langle \mathsf{m} \rangle \Big]^{\tau}, \Big[_{\mathsf{m}} \mathsf{p} \cdot (\overline{\mathsf{a}_0}\overline{\mathsf{b}_0} \cdot \mathsf{p})^* \cdot \overline{\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}} \cdot \langle_1 \mathsf{f}^1 \rangle \cdot \mathsf{q} \cdot \mathsf{r} \cdot \mathsf{s} \Big]^{\tau}, \Big[_{\mathsf{f}} \overline{\mathsf{a}\mathsf{b}} \cdot \mathsf{t} + \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot (\overline{\mathsf{a}\mathsf{b}} \cdot \mathsf{u} + \overline{\overline{\mathsf{a}\mathsf{b}}} \cdot \\ & \langle_2 \mathsf{f}^2 \rangle \cdot \mathsf{v} \cdot \mathsf{w}) \Big]^{\tau} \Big|\!\Big)^{\tau} \cdot \mathsf{m} \rangle \\ = \quad & \{\!\{ \text{ Equation (5.24) \& Remark: no valid derivation can } \textit{start} \text{ with } \tau \ \}\!\} \end{aligned}$$

$$
\begin{aligned}
& \langle \mathsf{m}\cdot\mathsf{p}\cdot(\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}\cdot\mathsf{p})^*\cdot\overline{\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}}\cdot\langle \mathsf{f}_1\cdot(\!|_{\mathsf{f}}\,[\![_{\mathsf{m}'}\langle \mathsf{m}\rangle]\!]^{\tau},\ [\![_{\mathsf{m}}\,\mathsf{p}\cdot(\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}\cdot\mathsf{p})^*\cdot\overline{\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}}\cdot\langle{}_1\mathsf{f}^1\rangle\cdot\mathsf{q}\cdot\mathsf{r}\cdot\mathsf{s}]\!]^{\tau},\\
& \qquad\qquad [\![_{\mathsf{f}}\,\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot(\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{u}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot\langle{}_2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})]\!]^{\tau}\,|\!)^{\tau}\cdot\mathsf{f}_1\rangle\cdot\mathsf{q}\cdot\mathsf{r}\cdot\mathsf{s}\cdot\mathsf{m}\rangle\\
= \quad & \{\!\{\ \text{Remove unused metablocks}\ \}\!\}\\
& \langle \mathsf{m}\cdot\mathsf{p}\cdot(\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}\cdot\mathsf{p})^*\cdot\overline{\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}}\cdot\langle \mathsf{f}_1\cdot(\!|_{\mathsf{f}}\,[\![_{\mathsf{f}}\,\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot(\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{u}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot\langle{}_2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})]\!]^{\tau}\,|\!)^{\tau}\cdot\mathsf{f}_1\rangle\cdot\mathsf{q}\cdot\mathsf{r}\cdot\mathsf{s}\cdot\mathsf{m}\rangle\\
= \quad & \{\!\{\ \text{Boolean algebra \& Hypotheses for } \langle \mathsf{f}_1\ \}\!\}\\
& \langle \mathsf{m}\cdot\mathsf{p}\cdot(\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}\cdot\mathsf{p})^*\cdot\langle \mathsf{f}_1\cdot\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot(\!|_{\mathsf{f}}\,[\![_{\mathsf{f}}\,\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot(\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{u}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot\langle{}_2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})]\!]^{\tau}\,|\!)^{\tau}\cdot\mathsf{f}_1\rangle\cdot\mathsf{q}\cdot\mathsf{r}\cdot\mathsf{s}\cdot\mathsf{m}\rangle\\
= \quad & \{\!\{\ \text{Equations (6.8), (6.9) and (6.10)}\ \}\!\}\\
& \langle \mathsf{m}\cdot\mathsf{p}\cdot(\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}\cdot\mathsf{p})^*\cdot\langle \mathsf{f}_1\cdot\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot(\!|_{\mathsf{f}}\,[\![_{\mathsf{f}}\,\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{u}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot\langle{}_2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w}]\!]^{\tau}\,|\!)^{\tau}\cdot\mathsf{f}_1\rangle\cdot\mathsf{q}\cdot\mathsf{r}\cdot\mathsf{s}\cdot\mathsf{m}\rangle\\
= \quad & \{\!\{\ \text{Add unused metablocks}\ \}\!\}\\
& \langle \mathsf{m}\cdot\mathsf{p}\cdot(\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}\cdot\mathsf{p})^*\cdot\langle \mathsf{f}_1\cdot\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot(\!|_{\mathsf{f}}\,[\![_{\mathsf{m}'}\langle \mathsf{m}\rangle]\!]^{\tau},\ [\![_{\mathsf{m}}\,\mathsf{p}\cdot(\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}\cdot\mathsf{p})^*\cdot\overline{\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}}\cdot\langle{}_1\mathsf{f}^1\rangle\cdot\mathsf{q}\cdot\mathsf{r}\cdot\mathsf{s}]\!]^{\tau},\\
& \qquad\qquad [\![_{\mathsf{f}}\,\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{u}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot\langle{}_2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w}]\!]^{\tau}\,|\!)^{\tau}\cdot\mathsf{f}_1\rangle\cdot\mathsf{q}\cdot\mathsf{r}\cdot\mathsf{s}\cdot\mathsf{m}\rangle\\
= \quad & \{\!\{\ \text{Boolean algebra \& Hypotheses for } \langle \mathsf{f}_1\ \}\!\}\\
& \langle \mathsf{m}\cdot\mathsf{p}\cdot(\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}\cdot\mathsf{p})^*\cdot\overline{\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}}\cdot\langle \mathsf{f}_1\cdot(\!|_{\mathsf{f}}\,[\![_{\mathsf{m}'}\langle \mathsf{m}\rangle]\!]^{\tau},\ [\![_{\mathsf{m}}\,\mathsf{p}\cdot(\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}\cdot\mathsf{p})^*\cdot\overline{\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}}\cdot\langle{}_1\mathsf{f}^1\rangle\cdot\mathsf{q}\cdot\mathsf{r}\cdot\mathsf{s}]\!]^{\tau},\\
& \qquad\qquad [\![_{\mathsf{f}}\,\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{u}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot\langle{}_2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w}]\!]^{\tau}\,|\!)^{\tau}\cdot\mathsf{f}_1\rangle\cdot\mathsf{q}\cdot\mathsf{r}\cdot\mathsf{s}\cdot\mathsf{m}\rangle\\
= \quad & \{\!\{\ \text{Equation (5.24) \& Remark: no valid derivation can } \textit{start} \text{ with } \tau\ \}\!\}\\
& \langle \mathsf{m}\cdot(\!|_{\mathsf{m}}\,[\![_{\mathsf{m}'}\langle \mathsf{m}\rangle]\!]^{\tau},\ [\![_{\mathsf{m}}\,\mathsf{p}\cdot(\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}\cdot\mathsf{p})^*\cdot\overline{\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}}\cdot\langle{}_1\mathsf{f}^1\rangle\cdot\mathsf{q}\cdot\mathsf{r}\cdot\mathsf{s}]\!]^{\tau},\ [\![_{\mathsf{f}}\,\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{u}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot\langle{}_2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w}]\!]^{\tau}\,|\!)^{\tau}\cdot\mathsf{m}\rangle\\
= \quad & \{\!\{\ \text{Equation (5.24) \& Remark: no valid derivation can } \textit{start} \text{ with } \tau\ \}\!\}\\
& (\!|_{\mathsf{m}'}\,[\![_{\mathsf{m}'}\langle \mathsf{m}\rangle]\!]^{\tau},\ [\![_{\mathsf{m}}\,\mathsf{p}\cdot(\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}\cdot\mathsf{p})^*\cdot\overline{\overline{\mathsf{a}_0}\overline{\mathsf{b}_0}}\cdot\langle{}_1\mathsf{f}^1\rangle\cdot\mathsf{q}\cdot\mathsf{r}\cdot\mathsf{s}]\!]^{\tau},\ [\![_{\mathsf{f}}\,\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{u}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot\langle{}_2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w}]\!]^{\tau}\,|\!)^{\tau}
\end{aligned}
$$

**Proof of** (6.8)**:** By the definition of metablock, it suffices to prove that

$$
\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot(\!|_{\mathsf{f}\,\mathsf{f}}\,[\![\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot(\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{u}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot\langle{}_2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})]\!]^{\tau}\,|\!)^{\tau} \quad = \quad (\!|_{\mathsf{f}\,\mathsf{f}}\,[\![\overline{\overline{\mathsf{a}\mathsf{b}}}]\!]^{\mathsf{g}},\,[\![_{\mathsf{g}}\,\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot(\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{u}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot\langle{}_2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})]\!]^{\tau}\,|\!)^{\tau} \ .
$$

By (5.24), it suffices to prove that

$$
\begin{aligned}
& \overline{\overline{\mathsf{a}\mathsf{b}}}\cdot(\!|_{\mathsf{f}\,\mathsf{f}}\,[\![\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot(\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{u}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot\langle{}_2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})]\!]^{\tau}\,|\!)^{\tau}\\
= \ & \overline{\overline{\mathsf{a}\mathsf{b}}}\cdot(\!|_{\mathsf{g}\,\mathsf{f}}\,[\![\overline{\overline{\mathsf{a}\mathsf{b}}}]\!]^{\mathsf{g}},\,[\![_{\mathsf{g}}\,\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot(\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{u}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot\langle{}_2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})]\!]^{\tau}\,|\!)^{\tau} \ .
\end{aligned}
$$

By Kleene algebra, it suffices to prove that

$$
\begin{aligned}
& (\!|_{\mathsf{f}\,\mathsf{f}}\,[\![\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot(\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{u}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot\langle{}_2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})]\!]^{\tau}\,|\!)^{\tau}\\
= \ & (\!|_{\mathsf{g}\,\mathsf{f}}\,[\![\overline{\overline{\mathsf{a}\mathsf{b}}}]\!]^{\mathsf{g}},\,[\![_{\mathsf{g}}\,\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot(\overline{\mathsf{a}\mathsf{b}}\cdot\mathsf{u}+\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot\langle{}_2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})]\!]^{\tau}\,|\!)^{\tau} \ .
\end{aligned}
\tag{6.11}
$$

The case $\geqslant$ of (6.11) is easy. We first show that

$$\begin{aligned} &\Big(\!\Big[\begin{smallmatrix}\mathsf{f}\\ \mathsf{f}\end{smallmatrix}\Big[\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}}\overline{\mathsf{b}}}\cdot(\mathsf{a}\mathsf{b}\cdot\mathsf{u}+\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\langle{}^2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})\Big]^{\tau\,\tau}\Big]\!\Big) \\ &= \Big(\!\Big[\begin{smallmatrix}\mathsf{f}\\ \mathsf{f}\end{smallmatrix}\Big[1\cdot(\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}}\overline{\mathsf{b}}}\cdot(\mathsf{a}\mathsf{b}\cdot\mathsf{u}+\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\langle{}^2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w}))\Big]^{\tau\,\tau}\Big]\!\Big) \quad . \end{aligned} \tag{6.12}$$

By (5.25), it suffices to prove that

$$\begin{aligned} &\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}}\overline{\mathsf{b}}}(\mathsf{a}\mathsf{b}\cdot\mathsf{u}+\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\langle\mathsf{f}_2\cdot\Big(\!\Big[\begin{smallmatrix}\mathsf{f}\\ \mathsf{f}\end{smallmatrix}\Big[1\cdot(\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}}\overline{\mathsf{b}}}\cdot(\mathsf{a}\mathsf{b}\cdot\mathsf{u}+\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\langle{}^2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w}))\Big]^{\tau\,\tau}\Big]\!\Big)\cdot\mathsf{f}_2\rangle\cdot\mathsf{v}\cdot\mathsf{w}) \\ &= 1\cdot\Big(\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}}\overline{\mathsf{b}}}(\mathsf{a}\mathsf{b}\cdot\mathsf{u}+\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\langle\mathsf{f}_2\cdot\Big(\!\Big[\begin{smallmatrix}\mathsf{f}\\ \mathsf{f}\end{smallmatrix}\Big[1\cdot(\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}}\overline{\mathsf{b}}}\cdot(\mathsf{a}\mathsf{b}\cdot\mathsf{u}+\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\langle{}^2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w}))\Big]^{\tau\,\tau}\Big]\!\Big)\cdot\mathsf{f}_2\rangle \\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\cdot\mathsf{v}\cdot\mathsf{w})\Big) \quad . \end{aligned}$$

This is trivial by Kleene algebra.

By the definition of metablock, equation (6.12) becomes

$$\begin{aligned} &\Big(\!\Big[\begin{smallmatrix}\mathsf{f}\\ \mathsf{f}\end{smallmatrix}\Big[\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}}\overline{\mathsf{b}}}\cdot(\mathsf{a}\mathsf{b}\cdot\mathsf{u}+\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\langle{}^2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})\Big]^{\tau\,\tau}\Big]\!\Big) \\ &= \Big(\!\Big[\begin{smallmatrix}\mathsf{f}\\ \mathsf{f}\end{smallmatrix}\Big[1\Big]^{\mathsf{g}},\ \Big[\begin{smallmatrix}\mathsf{g}\end{smallmatrix}\Big[\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}}\overline{\mathsf{b}}}\cdot(\mathsf{a}\mathsf{b}\cdot\mathsf{u}+\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\langle{}^2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})\Big]^{\tau\,\tau}\Big]\!\Big) \quad . \end{aligned}$$

By (5.24) and Kleene algebra, the previous equation becomes

$$\begin{aligned} &\Big(\!\Big[\begin{smallmatrix}\mathsf{f}\\ \mathsf{f}\end{smallmatrix}\Big[\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}}\overline{\mathsf{b}}}\cdot(\mathsf{a}\mathsf{b}\cdot\mathsf{u}+\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\langle{}^2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})\Big]^{\tau\,\tau}\Big]\!\Big) \\ &= \Big(\!\Big[\begin{smallmatrix}\mathsf{g}\\ \mathsf{f}\end{smallmatrix}\Big[1\Big]^{\mathsf{g}},\ \Big[\begin{smallmatrix}\mathsf{g}\end{smallmatrix}\Big[\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}}\overline{\mathsf{b}}}\cdot(\mathsf{a}\mathsf{b}\cdot\mathsf{u}+\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\langle{}^2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})\Big]^{\tau\,\tau}\Big]\!\Big) \quad . \end{aligned}$$

So, to prove the case $\geqslant$ of (6.11), it suffices to prove that

$$\begin{aligned} &\Big(\!\Big[\begin{smallmatrix}\mathsf{g}\\ \mathsf{f}\end{smallmatrix}\Big[\overline{\overline{\mathsf{a}}\overline{\mathsf{b}}}\Big]^{\mathsf{g}},\Big[\begin{smallmatrix}\mathsf{g}\end{smallmatrix}\Big[\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}}\overline{\mathsf{b}}}\cdot(\mathsf{a}\mathsf{b}\cdot\mathsf{u}+\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\langle{}^2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})\Big]^{\tau\,\tau}\Big]\!\Big) \\ &\leqslant \Big(\!\Big[\begin{smallmatrix}\mathsf{g}\\ \mathsf{f}\end{smallmatrix}\Big[1\Big]^{\mathsf{g}},\ \Big[\begin{smallmatrix}\mathsf{g}\end{smallmatrix}\Big[\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}}\overline{\mathsf{b}}}\cdot(\mathsf{a}\mathsf{b}\cdot\mathsf{u}+\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\langle{}^2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})\Big]^{\tau\,\tau}\Big]\!\Big) \quad . \end{aligned}$$

By (5.26), it suffices to prove that

$$\overline{\overline{a}\overline{b}} \quad \leqslant \quad 1 \quad .$$

This is trivial by Boolean algebra.

We now prove the case $\leqslant$ of (6.11). To facilitate the proof, we define

$$\begin{aligned} \mathcal{B}_1 &:= \Big[\begin{smallmatrix}\mathsf{f}\end{smallmatrix}\Big[\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}}\overline{\mathsf{b}}}\cdot(\mathsf{a}\mathsf{b}\cdot\mathsf{u}+\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\langle{}^2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})\Big]^{\tau} \quad , \\ \mathcal{C}_1 &:= \Big[\begin{smallmatrix}\mathsf{f}\end{smallmatrix}\Big[\overline{\overline{\mathsf{a}}\overline{\mathsf{b}}}\Big]^{\mathsf{g}},\Big[\begin{smallmatrix}\mathsf{g}\end{smallmatrix}\Big[\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\mathsf{t}+\overline{\overline{\mathsf{a}}\overline{\mathsf{b}}}\cdot(\mathsf{a}\mathsf{b}\cdot\mathsf{u}+\overline{\mathsf{a}}\overline{\mathsf{b}}\cdot\langle{}^2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})\Big]^{\tau} \quad . \end{aligned}$$

We use (5.10) with a substitution environment $\theta_{\mathcal{B}_1}$ defined by

$$\theta_{\mathcal{B}_1}(f,\tau) := \underset{g}{(\!|}\, \mathcal{C}_1 \,\overset{\tau}{|\!)} \ , \qquad \text{and} \qquad \theta_{\mathcal{B}_1}(\tau,\tau) := 0 \ .$$

So, it suffices to prove that

$$\mathsf{mb\_vpre}_{\theta_{\mathcal{B}_1}}(\overline{\mathsf{ab}}\cdot\mathsf{t} + \overline{\overline{\mathsf{ab}}}\cdot(\overline{\mathsf{ab}}\cdot\mathsf{u} + \overline{\overline{\mathsf{ab}}}\cdot(\langle \mathsf{f}_2 \underset{\mathsf{f}}{\downarrow} \overset{\tau}{\uparrow} \mathsf{f}_2\rangle\cdot\mathsf{v}\cdot\mathsf{w})) \quad \leqslant \quad \theta_{\mathcal{B}_1}(\mathsf{f},\tau) \ , \tag{6.13}$$

$$\mathsf{mb\_vpre}_{\theta_{\mathcal{B}_1}}(\overline{\mathsf{ab}}\cdot\mathsf{t} + \overline{\overline{\mathsf{ab}}}\cdot(\overline{\mathsf{ab}}\cdot\mathsf{u} + \overline{\overline{\mathsf{ab}}}\cdot(\langle \mathsf{f}_2 \downarrow_{\mathsf{f}} \uparrow^{\tau} \mathsf{f}_2\rangle\cdot\mathsf{v}\cdot\mathsf{w}))\cdot\theta_{\mathcal{B}_1}(\tau,\tau) \leqslant \theta_{\mathcal{B}_1}(\mathsf{f},\tau) \ . \tag{6.14}$$

Inequation (6.14) is trivial by the fact that $\theta_{\mathcal{B}_1}(\tau,\tau) = 0$ and by Kleene algebra. We now prove (6.13). By the definitions of $\mathsf{mb\_vpre}_{\theta_{\mathcal{B}_1}}$ and $\theta_{\mathcal{B}_1}$, it suffices to prove that

$$\overline{\mathsf{ab}}\cdot\mathsf{t} + \overline{\overline{\mathsf{ab}}}(\overline{\mathsf{ab}}\cdot\mathsf{u} + \overline{\overline{\mathsf{ab}}}\cdot\langle \mathsf{f}_2 \cdot \underset{g}{(\!|}\, \mathcal{C}_1 \,\overset{\tau}{|\!)}\cdot \mathsf{f}_2\rangle\cdot\mathsf{v}\cdot\mathsf{w}) \quad \leqslant \quad \underset{g}{(\!|}\, \mathcal{C}_1 \,\overset{\tau}{|\!)} \ .$$

We prove it.

$$\begin{aligned}
& \overline{\mathsf{ab}}\cdot\mathsf{t} + \overline{\overline{\mathsf{ab}}}(\overline{\mathsf{ab}}\cdot\mathsf{u} + \overline{\overline{\mathsf{ab}}}\cdot\langle \mathsf{f}_2 \cdot (\!|_{\mathsf{g}}\, \mathcal{C}_1 \,|\!)^{\tau}\cdot \mathsf{f}_2\rangle\cdot\mathsf{v}\cdot\mathsf{w}) \\
= & \qquad \{\!\{ \text{ Distributivity of } \cdot \text{ on } + \ \}\!\} \\
& \overline{\mathsf{ab}}\cdot\mathsf{t} + (\overline{\overline{\mathsf{ab}}}\cdot\overline{\mathsf{ab}}\cdot\mathsf{u} + \overline{\overline{\mathsf{ab}}}\cdot\overline{\mathsf{ab}}\cdot\langle \mathsf{f}_2 \cdot (\!|_{\mathsf{g}}\, \mathcal{C}_1 \,|\!)^{\tau}\cdot \mathsf{f}_2\rangle\cdot\mathsf{v}\cdot\mathsf{w}) \\
= & \qquad \{\!\{ \text{ Boolean algebra: idempotency of tests \& Distributivity of } \cdot \text{ on } + \ \}\!\} \\
& \overline{\mathsf{ab}}\cdot\mathsf{t} + \overline{\overline{\mathsf{ab}}}(\overline{\mathsf{ab}}\cdot\mathsf{u} + \overline{\overline{\mathsf{ab}}}\cdot\overline{\mathsf{ab}}\cdot\langle \mathsf{f}_2 \cdot (\!|_{\mathsf{g}}\, \mathcal{C}_1 \,|\!)^{\tau}\cdot \mathsf{f}_2\rangle\cdot\mathsf{v}\cdot\mathsf{w}) \\
= & \qquad \{\!\{ \text{ Boolean algebra } \}\!\} \\
& \overline{\mathsf{ab}}\cdot\mathsf{t} + \overline{\overline{\mathsf{ab}}}(\overline{\mathsf{ab}}\cdot\mathsf{u} + \overline{\overline{\mathsf{ab}}}\cdot(\mathsf{ab}+\mathsf{a}\overline{\mathsf{b}})\cdot\langle \mathsf{f}_2 \cdot (\!|_{\mathsf{g}}\, \mathcal{C}_1 \,|\!)^{\tau}\cdot \mathsf{f}_2\rangle\cdot\mathsf{v}\cdot\mathsf{w}) \\
= & \qquad \{\!\{ \text{ Hypotheses: } \mathsf{ab}\cdot\langle \mathsf{f}_2 = \langle \mathsf{f}_2\cdot(\mathsf{ab}+\mathsf{a}\overline{\mathsf{b}}) \text{ and } \mathsf{a}\overline{\mathsf{b}}\cdot\langle \mathsf{f}_2 = \langle \mathsf{f}_2\cdot\overline{\mathsf{ab}}, \text{ so, by Kleene} \\
& \quad \text{algebra, } (\mathsf{ab}+\mathsf{a}\overline{\mathsf{b}})\cdot\langle \mathsf{f}_2 = \langle \mathsf{f}_2\cdot(\mathsf{ab}+\mathsf{a}\overline{\mathsf{b}}+\overline{\mathsf{ab}}). \ \}\!\} \\
& \overline{\mathsf{ab}}\cdot\mathsf{t} + \overline{\overline{\mathsf{ab}}}(\overline{\mathsf{ab}}\cdot\mathsf{u} + \overline{\overline{\mathsf{ab}}}\cdot\langle \mathsf{f}_2 \cdot(\mathsf{ab}+\mathsf{a}\overline{\mathsf{b}}+\overline{\mathsf{ab}})\cdot (\!|_{\mathsf{g}}\, \mathcal{C}_1 \,|\!)^{\tau}\cdot \mathsf{f}_2\rangle\cdot\mathsf{v}\cdot\mathsf{w}) \\
= & \qquad \{\!\{ \text{ Boolean algebra } \}\!\} \\
& \overline{\mathsf{ab}}\cdot\mathsf{t} + \overline{\overline{\mathsf{ab}}}(\overline{\mathsf{ab}}\cdot\mathsf{u} + \overline{\overline{\mathsf{ab}}}\cdot\langle \mathsf{f}_2 \cdot\overline{\overline{\mathsf{ab}}}\cdot (\!|_{\mathsf{g}}\, \mathcal{C}_1 \,|\!)^{\tau}\cdot \mathsf{f}_2\rangle\cdot\mathsf{v}\cdot\mathsf{w}) \\
= & \qquad \{\!\{ \text{ Law (5.24) } \}\!\} \\
& \overline{\mathsf{ab}}\cdot\mathsf{t} + \overline{\overline{\mathsf{ab}}}(\overline{\mathsf{ab}}\cdot\mathsf{u} + \overline{\overline{\mathsf{ab}}}\cdot\langle \mathsf{f}_2 \cdot (\!|_{\mathsf{f}}\, \mathcal{C}_1 \,|\!)^{\tau}\cdot \mathsf{f}_2\rangle\cdot\mathsf{v}\cdot\mathsf{w}) \\
\leqslant & \qquad \{\!\{ \text{ Law (5.1) } \}\!\} \\
& (\!|_{\mathsf{g}}\, \mathcal{C}_1 \,|\!)^{\tau}
\end{aligned}$$

**Proof of (6.9):** By (5.25), it suffices to prove that

$$\begin{aligned}
& \overline{\overline{\mathsf{ab}}}\cdot(\overline{\mathsf{ab}}\cdot\mathsf{t} + \overline{\overline{\mathsf{ab}}}(\overline{\mathsf{ab}}\cdot\mathsf{u} + \overline{\overline{\mathsf{ab}}}\cdot\langle \mathsf{f}_2 \cdot \underset{\mathsf{f}\ \mathsf{f}}{(\!|}\, [\, \overline{\overline{\mathsf{ab}}}\cdot(\overline{\mathsf{ab}}\cdot\mathsf{u} + \overline{\overline{\mathsf{ab}}}\cdot\langle^2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})\,] \,\overset{\tau\ \tau}{|\!)}\cdot \mathsf{f}_2\rangle\cdot\mathsf{v}\cdot\mathsf{w})) \\
& = \overline{\overline{\mathsf{ab}}}\cdot(\overline{\mathsf{ab}}\cdot\mathsf{u} + \overline{\overline{\mathsf{ab}}}\cdot\langle \mathsf{f}_2 \cdot \underset{\mathsf{f}\ \mathsf{f}}{(\!|}\, [\, \overline{\overline{\mathsf{ab}}}\cdot(\overline{\mathsf{ab}}\cdot\mathsf{u} + \overline{\overline{\mathsf{ab}}}\cdot\langle^2\mathsf{f}^2\rangle\cdot\mathsf{v}\cdot\mathsf{w})\,] \,\overset{\tau\ \tau}{|\!)}\cdot \mathsf{f}_2\rangle\cdot\mathsf{v}\cdot\mathsf{w}) \ .
\end{aligned}$$

We prove it.

$$
\begin{aligned}
& \overline{\overline{\mathsf{ab}}} \cdot (\overline{\mathsf{ab}} \cdot \mathsf{t} + \overline{\overline{\mathsf{ab}}}(\overline{\mathsf{ab}} \cdot \mathsf{u} + \overline{\mathsf{ab}} \cdot \langle \mathsf{f}_2 \cdot (\!|_{\mathsf{f}} \, [_{\mathsf{f}} \, \overline{\overline{\mathsf{ab}}} \cdot (\overline{\mathsf{ab}} \cdot \mathsf{u} + \overline{\mathsf{ab}} \cdot \langle {}_2\mathsf{f}^2 \rangle \cdot \mathsf{v} \cdot \mathsf{w}) \, ]^\tau \, |\!)^\tau \cdot \mathsf{f}_2^{\rangle} \cdot \mathsf{v} \cdot \mathsf{w})) \\
= & \quad \{\!\{ \text{ Distributivity of } \cdot \text{ on } + \text{ } \}\!\} \\
& \overline{\overline{\mathsf{ab}}} \cdot \overline{\mathsf{ab}} \cdot \mathsf{t} + \overline{\overline{\mathsf{ab}}} \cdot \overline{\overline{\mathsf{ab}}}(\overline{\mathsf{ab}} \cdot \mathsf{u} + \overline{\mathsf{ab}} \cdot \langle \mathsf{f}_2 \cdot (\!|_{\mathsf{f}} \, [_{\mathsf{f}} \, \overline{\overline{\mathsf{ab}}} \cdot (\overline{\mathsf{ab}} \cdot \mathsf{u} + \overline{\mathsf{ab}} \cdot \langle {}_2\mathsf{f}^2 \rangle \cdot \mathsf{v} \cdot \mathsf{w}) \, ]^\tau \, |\!)^\tau \cdot \mathsf{f}_2^{\rangle} \cdot \mathsf{v} \cdot \mathsf{w}) \\
= & \quad \{\!\{ \text{ Boolean algebra: contradiction of tests and idempotency of tests } \}\!\} \\
& 0 \cdot \mathsf{t} + \overline{\overline{\mathsf{ab}}}(\overline{\mathsf{ab}} \cdot \mathsf{u} + \overline{\mathsf{ab}} \cdot \langle \mathsf{f}_2 \cdot (\!|_{\mathsf{f}} \, [_{\mathsf{f}} \, \overline{\overline{\mathsf{ab}}} \cdot (\overline{\mathsf{ab}} \cdot \mathsf{u} + \overline{\mathsf{ab}} \cdot \langle {}_2\mathsf{f}^2 \rangle \cdot \mathsf{v} \cdot \mathsf{w}) \, ]^\tau \, |\!)^\tau \cdot \mathsf{f}_2^{\rangle} \cdot \mathsf{v} \cdot \mathsf{w}) \\
= & \quad \{\!\{ \text{ Zero of } \cdot \text{ \& Identity of } + \text{ } \}\!\} \\
& \overline{\overline{\mathsf{ab}}} \cdot (\overline{\mathsf{ab}} \cdot \mathsf{u} + \overline{\mathsf{ab}} \cdot \langle \mathsf{f}_2 \cdot (\!|_{\mathsf{f}} \, [_{\mathsf{f}} \, \overline{\overline{\mathsf{ab}}} \cdot (\overline{\mathsf{ab}} \cdot \mathsf{u} + \overline{\mathsf{ab}} \cdot \langle {}_2\mathsf{f}^2 \rangle \cdot \mathsf{v} \cdot \mathsf{w}) \, ]^\tau \, |\!)^\tau \cdot \mathsf{f}_2^{\rangle} \cdot \mathsf{v} \cdot \mathsf{w})
\end{aligned}
$$

**Proof of** (6.10)**:** The proof of the case $\leqslant$ is similar to the case $\geqslant$ of (6.8). Then, it remains to prove the case $\geqslant$. We have to prove that

$$
\overline{\overline{\mathsf{ab}}} \cdot (\!|_{\mathsf{f}\,\mathsf{f}} \, [\, \overline{\mathsf{ab}} \cdot \mathsf{u} + \overline{\mathsf{ab}} \cdot \langle {}_2\mathsf{f}^2 \rangle \cdot \mathsf{v} \cdot \mathsf{w} \, ]^{\tau\,\tau} \, |\!) \quad \leqslant \quad (\!|_{\mathsf{f}\,\mathsf{f}} \, [\, \overline{\overline{\mathsf{ab}}} \cdot (\overline{\mathsf{ab}} \cdot \mathsf{u} + \overline{\mathsf{ab}} \cdot \langle {}_2\mathsf{f}^2 \rangle \cdot \mathsf{v} \cdot \mathsf{w}) \, ]^{\tau\,\tau} \, |\!) \ .
$$

Since, by simple Kleene algebra with tests reasoning, for all expressions $p$

$$
\begin{aligned}
& \overline{\overline{\mathsf{ab}}} \cdot (\overline{\mathsf{ab}} \cdot \mathsf{u} + \overline{\overline{\mathsf{ab}}} \cdot \langle \mathsf{f}_2 \cdot p \cdot \mathsf{f}_2 \rangle \cdot \mathsf{v} \cdot \mathsf{w}) \\
= \ & \overline{\overline{\mathsf{ab}}} \cdot \overline{\mathsf{ab}} \cdot \overline{\overline{\mathsf{ab}}} \cdot \overline{\overline{\mathsf{ab}}} \cdot \mathsf{u} \cdot 1 \\
& + \ \overline{\overline{\mathsf{ab}}} \cdot \overline{\overline{\mathsf{ab}}} \cdot (\mathsf{ab} + \mathsf{a}\overline{\mathsf{b}}) \cdot (\mathsf{ab} + \mathsf{a}\overline{\mathsf{b}}) \cdot \langle \mathsf{f}_2 \cdot p \cdot \mathsf{f}_2 \rangle \cdot 1 \cdot 1 \cdot \mathsf{v} \cdot 1 \cdot 1 \cdot \mathsf{w} \cdot 1 \ ,
\end{aligned}
$$

then, by (5.25), it suffices to prove that

$$
\begin{aligned}
& \overline{\overline{\mathsf{ab}}} \cdot (\!|_{\mathsf{f}\,\mathsf{f}} \, [\, \overline{\mathsf{ab}} \cdot \mathsf{u} + \overline{\overline{\mathsf{ab}}} \cdot \langle {}_2\mathsf{f}^2 \rangle \cdot \mathsf{v} \cdot \mathsf{w} \, ]^{\tau\,\tau} \, |\!) \\
\leqslant \ & (\!|_{\mathsf{f}\,\mathsf{f}} \, [\, \overline{\overline{\mathsf{ab}}} \cdot \overline{\mathsf{ab}} \cdot \overline{\overline{\mathsf{ab}}} \cdot \overline{\overline{\mathsf{ab}}} \cdot \mathsf{u} \cdot 1 \\
& \qquad + \overline{\overline{\mathsf{ab}}} \cdot \overline{\overline{\mathsf{ab}}} \cdot (\mathsf{ab} + \mathsf{a}\overline{\mathsf{b}}) \cdot (\mathsf{ab} + \mathsf{a}\overline{\mathsf{b}}) \cdot \langle {}_2\mathsf{f}^2 \rangle \cdot 1 \cdot 1 \cdot \mathsf{v} \cdot 1 \cdot 1 \cdot \mathsf{w} \cdot 1 \, ]^{\tau\,\tau} \, |\!) \ .
\end{aligned}
$$

By the definition of metablock, it suffices to prove that

$$
\begin{aligned}
& \overline{\overline{\mathsf{ab}}} \cdot (\!|_{\mathsf{f}\,\mathsf{f}} \, [\, \overline{\mathsf{ab}} \,]^{\mathsf{g}_1}_{\mathsf{g}_1}, \, [\, \mathsf{u} \,]^{\tau}_{\mathsf{g}_1}, \, [\, \overline{\overline{\mathsf{ab}}} \,]^{\mathsf{g}_2}_{\mathsf{f}}, \, [\, \langle {}_2\mathsf{f}^2 \rangle \,]^{\mathsf{g}_3}_{\mathsf{g}_2}, \, [\, \mathsf{v} \,]^{\mathsf{g}_4}_{\mathsf{g}_3}, \, [\, \mathsf{w} \,]^{\tau\,\tau}_{\mathsf{g}_4} \, |\!) \\
& \leqslant (\!|_{\mathsf{f}\,\mathsf{f}} \, [\, \overline{\overline{\mathsf{ab}}} \cdot \overline{\mathsf{ab}} \cdot \overline{\overline{\mathsf{ab}}} \,]^{\mathsf{g}_1}_{\mathsf{g}_1}, \, [\, \overline{\overline{\mathsf{ab}}} \cdot \mathsf{u} \cdot 1 \,]^{\tau}_{\mathsf{g}_1}, \, [\, \overline{\overline{\mathsf{ab}}} \cdot \overline{\overline{\mathsf{ab}}} \cdot (\mathsf{ab} + \mathsf{a}\overline{\mathsf{b}}) \,]^{\mathsf{g}_2}_{\mathsf{f}}, \\
& \qquad [\, (\mathsf{ab} + \mathsf{a}\overline{\mathsf{b}}) \cdot \langle {}_2\mathsf{f}^2 \rangle \cdot 1 \,]^{\mathsf{g}_3}_{\mathsf{g}_2}, \, [\, 1 \cdot \mathsf{v} \cdot 1 \,]^{\mathsf{g}_4}_{\mathsf{g}_3}, \, [\, 1 \cdot \mathsf{w} \cdot 1 \,]^{\tau\,\tau}_{\mathsf{g}_4} \, |\!) \ .
\end{aligned}
\tag{6.15}
$$

To prove (6.15), by (5.43), it suffices to prove that

- for $\lfloor_{\mathsf{f}}\, \overline{\mathsf{a}}\mathsf{b}\,\rfloor^{\mathsf{g}_1}$,

$$\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot\overline{\mathsf{a}}\mathsf{b} \quad\leqslant\quad \overline{\mathsf{a}}\mathsf{b}\cdot\overline{\overline{\mathsf{a}\mathsf{b}}}\ , \tag{6.16}$$

- for $\lfloor_{\mathsf{g}_1}\, \mathsf{u}\,\rfloor^{\tau}$,

$$\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot\mathsf{u} \quad\leqslant\quad \mathsf{u}\cdot 1\ , \tag{6.17}$$

- for $\lfloor_{\mathsf{f}}\, \overline{\overline{\mathsf{a}\mathsf{b}}}\,\rfloor^{\mathsf{g}_2}$,

$$\overline{\overline{\mathsf{a}\mathsf{b}}}\cdot\overline{\mathsf{a}\mathsf{b}} \quad\leqslant\quad \overline{\mathsf{a}\mathsf{b}}\cdot(\mathsf{a}\mathsf{b}+\mathsf{a}\overline{\mathsf{b}})\ , \tag{6.18}$$

- for $\lfloor_{\mathsf{g}_3}\, \mathsf{v}\,\rfloor^{\mathsf{g}_4}$,

$$1\cdot\mathsf{v} \quad\leqslant\quad \mathsf{v}\cdot 1\ , \tag{6.19}$$

- for $\lfloor_{\mathsf{g}_4}\, \mathsf{w}\,\rfloor^{\tau}$,

$$1\cdot\mathsf{w} \quad\leqslant\quad \mathsf{w}\cdot 1\ , \tag{6.20}$$

- for $\lfloor_{\mathsf{g}_2}\, {}^{\langle_2}\mathsf{f}^{2\rangle}\,\rfloor^{\mathsf{g}_3}$,

$$(\mathsf{a}\mathsf{b}+\mathsf{a}\overline{\mathsf{b}})\cdot{}^{\langle}\mathsf{f}_2 \quad\leqslant\quad {}^{\langle}\mathsf{f}_2\cdot\overline{\overline{\mathsf{a}\mathsf{b}}}\ , \tag{6.21}$$

$$1\cdot\mathsf{f}_2^{\rangle} \quad\leqslant\quad \mathsf{f}_2^{\rangle}\cdot 1\ . \tag{6.22}$$

Each of these proofs is easy. The proof of (6.16) is trivial by the commutativity of tests of Boolean algebra. The proofs of (6.17), (6.19), (6.20) and (6.22) are trivial by Boolean algebra. The proof of (6.18) is trivial by Boolean algebra. For (6.21), by hypotheses $\mathsf{a}\mathsf{b}\cdot{}^{\langle}\mathsf{f}_2 = {}^{\langle}\mathsf{f}_2\cdot(\mathsf{a}\mathsf{b}+\mathsf{a}\overline{\mathsf{b}})$ and $\mathsf{a}\overline{\mathsf{b}}\cdot{}^{\langle}\mathsf{f}_2 = {}^{\langle}\mathsf{f}_2\cdot\overline{\mathsf{a}}\mathsf{b}$, and by Kleene algebra, $(\mathsf{a}\mathsf{b}+\mathsf{a}\overline{\mathsf{b}})\cdot{}^{\langle}\mathsf{f}_2 = {}^{\langle}\mathsf{f}_2\cdot(\mathsf{a}\mathsf{b}+\mathsf{a}\overline{\mathsf{b}}+\overline{\mathsf{a}}\mathsf{b})$. So, by Boolean algebra, $(\mathsf{a}\mathsf{b}+\mathsf{a}\overline{\mathsf{b}})\cdot{}^{\langle}\mathsf{f}_2 = {}^{\langle}\mathsf{f}_2\cdot\overline{\overline{\mathsf{a}\mathsf{b}}}$.

#### Dead Function Elimination

A special case of interprocedural dead code elimination is the elimination of "unused" functions in the source code (there is no call to these functions in the source code). This elimination is trivial using VPKAT since it is an easy theorem of this algebraic system: any unreachable block can be removed from the list of blocks. One can see intuitively why by recalling the function $\mathsf{F}^*_{\mathcal{B}}$ of axiom (3.14) of page 43.

### 6.3.2 Inlining of Functions

The inlining of functions is an optimization that replaces a function call site with the body of the callee. This may improve the time performance of the program, but may

```
int main(void) {
  int x = 2;
  int* n;

  /* Inlining the first call to increment. */
  n = &x;
  *n += 1;
  /* Inlining the second call to increment. */
  n = &x;
  *n += 1;

  return 0; /*Success. */
}
```

Figure 6.11: Inlining of increment in the simple non-recursive program example.

also increase the size of the resulting program. Note that not every function can be inlined. For example, recursive functions may not always be inlined.

For example, the function increment of the program of Figure 6.9b can be inlined. There are different ways to handle the arguments of a function while inlining it. One easy way is to add an explicit assignment instruction before the body of the function. For the example program of Figure 6.9b, inlining would give the program of Figure 6.11.

Let $\mathsf{u}$ be the internal action $\mathsf{n} = \&\mathsf{x}$. Using the alphabets defined on page 138, the control flow of the previous program can be represented by the VPKA expression:

$$\langle\!\langle \underset{\mathsf{m}'}{[} \underset{\mathsf{m}'}{\langle \mathsf{m} \rangle^{\tau}} \underset{\mathsf{m}'}{]}, \underset{\mathsf{m}}{[} \mathsf{s} \cdot \mathsf{u} \cdot \mathsf{q} \cdot \mathsf{u} \cdot \mathsf{q} \cdot \mathsf{t} \underset{\mathsf{m}}{]^{\tau}} \rangle\!\rangle^{\tau} \ .$$

The formalization must rely on a compiler's trusted functionality that tells if a particular function can be inlined. When knowing if a function can be inlined, it becomes an assumption that can be expressed in VPKAT for this particular case. Then, the verification can be done for the entire program.

The assumption that the function increment can be inlined in the program of Figure 6.9b is just the equational hypothesis:

$$\langle \mathsf{f} \cdot \mathsf{q} \cdot \mathsf{f} \rangle \quad = \quad \mathsf{u} \cdot \mathsf{q} \ , \tag{6.23}$$

So, using the inlining hypothesis (6.23), we can prove that

$$\Big(\!\Big|\, [\langle \mathsf{m} \rangle]^{\tau}_{\mathsf{m}'\,\mathsf{m}'},\ [\mathsf{s}\cdot\langle \mathsf{f}\rangle\cdot\langle \mathsf{f}\rangle\cdot\mathsf{t}]^{\tau}_{\mathsf{m}},\ [\mathsf{q}]^{\tau\,\tau}_{\mathsf{f}} \,\Big|\!\Big) \quad = \quad \Big(\!\Big|\, [\langle \mathsf{m} \rangle]^{\tau}_{\mathsf{m}'\,\mathsf{m}'},\ [\mathsf{s}\cdot\mathsf{u}\cdot\mathsf{q}\cdot\mathsf{u}\cdot\mathsf{q}\cdot\mathsf{t}]^{\tau\,\tau}_{\mathsf{m}} \,\Big|\!\Big) \ .$$

By adding an unused block, it suffices to prove that

$$\Big(\!\Big|\, [\langle \mathsf{m} \rangle]^{\tau}_{\mathsf{m}'\,\mathsf{m}'},\ [\mathsf{s}\cdot\langle \mathsf{f}\rangle\cdot\langle \mathsf{f}\rangle\cdot\mathsf{t}]^{\tau}_{\mathsf{m}},\ [\mathsf{q}]^{\tau\,\tau}_{\mathsf{f}} \,\Big|\!\Big) \quad = \quad \Big(\!\Big|\, [\langle \mathsf{m} \rangle]^{\tau}_{\mathsf{m}'\,\mathsf{m}'},\ [\mathsf{s}\cdot\mathsf{u}\cdot\mathsf{q}\cdot\mathsf{u}\cdot\mathsf{q}\cdot\mathsf{t}]^{\tau}_{\mathsf{m}},\ [\mathsf{q}]^{\tau\,\tau}_{\mathsf{f}} \,\Big|\!\Big) \ .$$

By (5.25), it suffices to prove that

$$\begin{aligned} &\mathsf{s}\cdot\langle \mathsf{f}\cdot\Big(\!\Big|\, [\langle \mathsf{m} \rangle]^{\tau}_{\mathsf{f}\,\mathsf{m}'},\ [\mathsf{s}\cdot\mathsf{u}\cdot\mathsf{q}\cdot\mathsf{u}\cdot\mathsf{q}\cdot\mathsf{t}]^{\tau}_{\mathsf{m}},[\mathsf{q}]^{\tau\,\tau}_{\mathsf{f}} \,\Big|\!\Big)\cdot\mathsf{f}\rangle \\ &\quad\cdot\langle \mathsf{f}\cdot\Big(\!\Big|\, [\langle \mathsf{m} \rangle]^{\tau}_{\mathsf{f}\,\mathsf{m}'},\ [\mathsf{s}\cdot\mathsf{u}\cdot\mathsf{q}\cdot\mathsf{u}\cdot\mathsf{q}\cdot\mathsf{t}]^{\tau}_{\mathsf{m}},[\mathsf{q}]^{\tau\,\tau}_{\mathsf{f}} \,\Big|\!\Big)\cdot\mathsf{f}\rangle\cdot\mathsf{t} \\ &= \mathsf{s}\cdot\mathsf{u}\cdot\mathsf{q}\cdot\mathsf{u}\cdot\mathsf{q}\cdot\mathsf{t} \ . \end{aligned}$$

By (5.24) and the remark that no valid derivation can *start* with $\tau$, it suffices to prove that

$$\mathsf{s}\cdot\langle \mathsf{f}\cdot\mathsf{q}\cdot\mathsf{f}\rangle\cdot\langle \mathsf{f}\cdot\mathsf{q}\cdot\mathsf{f}\rangle\cdot\mathsf{t} \quad = \quad \mathsf{s}\cdot\mathsf{u}\cdot\mathsf{q}\cdot\mathsf{u}\cdot\mathsf{q}\cdot\mathsf{t} \ .$$

This is trivial by the inlining hypothesis (6.23).

### 6.3.3 Tail-Recursion Elimination

Tail-recursion elimination is an optimization that allows a compiler to rewrite a tail-recursive function[3] as a non-recursive function having an iterative form.

Like for inlining of functions, the formalization of a program in this situation relies on a compiler's trusted functionality that tells if a particular function is in tail-recursive form. When knowing it, it becomes an assumption for this particular case. The assumption is similar to the one generated for the inlining of functions. So, the verification can be done for the entire program.

### 6.3.4 Procedure Reordering

A common and easy interprocedural optimization is to reorder functions based on their call relationship. The central idea of this optimization is to minimize instruction cache thrashing by having the caller and the callee near each other in the program file.

[3] A function is tail-recursive if the only recursive calls it contains are tail-recursive. A function call is tail-recursive if there is nothing to do after the function returns except returning its value.

Procedure reordering is easy to deal with in VPKAT since it is only an application of the *swap blocks* laws (3.33) to (3.35).

### 6.3.5 Function Cloning

At first, cloning a function may seem a waste of memory. However, cloning a function allows a compiler to specialize it (and thus optimize it) for a specific subset of the call sites of this function. For example, it can be useful to remove unnecessary tests in a function for certain call sites (recall the interprocedural dead code elimination of Section 6.3.1). So, function cloning is useful in presence of other optimizations.

Correctness of function cloning is easy to show in VPKAT. It is just applying a theorem of VPKAT.

**Theorem 6.1** (Correctness of function cloning)**.** *Let $\mathcal{B}$ be a finite list of metablocks for a program. So, $\mathcal{B}$ contains only metablocks of the form $[\![_x\, p]\!]^\tau$. Let $[\![_f\, q]\!]^\tau$ be the metablock encoding of the function $f$ to be cloned. Let $\mathcal{C}$ be a list that contains $[\![_f\, q]\!]^\tau$ and all the metablocks called by it in $\mathcal{B}$ (recursively). Let $f'$ be a fresh label. Let $\mathcal{C}'$ be the list $\mathcal{C}$ in which every starting and calling label $f$ is replaced with $f'$. Let $\mathcal{D}$ be the list $\mathcal{B}$ without the metablocks of $\mathcal{C}$. Let $\mathcal{D}'$ be the list $\mathcal{D}$ in which* some calling labels *$f$ are replaced with $f'$. Then, for every starting label $g$ in $\mathcal{D}'$,*

$$(\!|\, \mathcal{B} \,|\!)^\tau_g \quad = \quad (\!|\, \mathcal{D}', \mathcal{C}, \mathcal{C}' \,|\!)^\tau_g \ . \tag{6.24}$$

*Proof.* For the case $\geqslant$ of (6.24), the proof is done by using the *substitution function* law (3.37) of VPKA. Let $h$ be a substitution function defined by $h(f') := f$ and by $h(v) := v$ for all labels $v$ of $\mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}')$. The proof is done by noting that $\widehat{h}(\mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}'))$, gives exactly $\mathsf{mb}(\mathcal{B})$ when the list $\widehat{h}(\mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}'))$ is shrinked with idempotency and swapping of blocks.

For the case $\leqslant$ of (6.24), first note that

$$(\!|\, \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \,|\!)^\tau_f \quad = \quad (\!|\, \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \,|\!)^\tau_{f'} \ . \tag{6.25}$$

By the definition of $\mathcal{C}$, $\mathcal{C}'$ and $\mathcal{D}'$, there are some unused blocks in these expressions. So, it suffices to prove that

$$(\!|\, \mathsf{mb}(\mathcal{C}) \,|\!)^\tau_f \quad = \quad (\!|\, \mathsf{mb}(\mathcal{C}') \,|\!)^\tau_{f'} \ .$$

Recall that $\mathsf{mb}(\mathcal{C}')$ differs from $\mathsf{mb}(\mathcal{C})$ only for the use of label $f'$ instead of $f$. So, the proof is trivial with the substitution function law of VPKA. For the case $\geqslant$, let $h_2$ be a substitution function defined by $h_2(f') := f$ and by $h_2(v) := v$ for the remaining labels. But, $\widehat{h_2}(\mathsf{mb}(\mathcal{C}'))$ is exactly $\mathsf{mb}(\mathcal{C})$. For the case $\leqslant$, use $h_2^{-1}$ as the substitution function.

We now prove the case $\leqslant$ of (6.24). Note that, for every starting label $g$ in $\mathcal{D}'$, $\mathsf{F}^*_{\mathsf{mb}(\mathcal{B})}(\{(g,\tau)\})$ contains only pairs of the form $(x,\tau)$ where $x$ is in $\mathsf{mb}(\mathcal{B})$. So, we use (3.14) with, for all labels $x$ in $\mathsf{mb}(\mathcal{B})$,

$$s_{(x,\tau)} := \underset{x}{(\!|}\, \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \,\underset{x}{|\!)}^{\tau} \quad .$$

So, by the definition of $\mathcal{B}$, it suffices to prove that, for all labels $u$ in $\mathsf{mb}(\mathcal{B})$,

$$\left( \bigwedge\ m \mid \underset{u}{[}\, m \,\underset{u}{]}^{\tau} \in \mathsf{mb}(\mathcal{B})^1 : m \quad \leqslant \quad \underset{u}{(\!|}\, \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \,\underset{u}{|\!)}^{\tau} \right) \ , \tag{6.26}$$

$$\left( \bigwedge\ m, v \mid \underset{u}{[}\, m \,\underset{u}{]}^{v} \in \mathsf{mb}(\mathcal{B})^1 : m \cdot \underset{v}{(\!|}\, \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \,\underset{v}{|\!)}^{\tau} \quad \leqslant \quad \underset{u}{(\!|}\, \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \,\underset{u}{|\!)}^{\tau} \right) \ , \tag{6.27}$$

$$\left( \bigwedge\ c, z, r \mid \underset{u}{[}\, c \underset{z}{\downarrow} \overset{\tau}{\uparrow} r \,\underset{u}{]}^{\tau} \in \mathsf{mb}(\mathcal{B})^2 : c \cdot \underset{z}{(\!|}\, \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \,\underset{z}{|\!)}^{\tau} \cdot r \leqslant \underset{u}{(\!|}\, \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \,\underset{u}{|\!)}^{\tau} \right) \ , \tag{6.28}$$

$$\left( \bigwedge\ c, z, r, v \mid \underset{u}{[}\, c \underset{z}{\downarrow} \overset{\tau}{\uparrow} r \,\underset{u}{]}^{v} \in \mathsf{mb}(\mathcal{B})^2 : c \cdot \underset{z}{(\!|}\, \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \,\underset{z}{|\!)}^{\tau} \cdot r \cdot \underset{v}{(\!|}\, \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \,\underset{v}{|\!)}^{\tau} \leqslant \underset{u}{(\!|}\, \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \,\underset{u}{|\!)}^{\tau} \right) \ . \tag{6.29}$$

We first prove (6.26). We suppose that there exists a unary block $\underset{u}{[}\, m \,\underset{u}{]}^{\tau}$ in $\mathsf{mb}(\mathcal{B})^1$ and we prove that

$$m \quad \leqslant \quad \underset{u}{(\!|}\, \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \,\underset{u}{|\!)}^{\tau} \quad .$$

By definition of $\mathcal{B}$, $\mathcal{C}$ and $\mathcal{D}'$, every unary block in $\mathcal{B}$ is in $\mathcal{C}$ or in $\mathcal{D}'$. So, the proof follows from axiom (3.2).

We now prove (6.27). We suppose that there exists a unary block $\underset{u}{[}\, m \,\underset{u}{]}^{v}$ in $\mathsf{mb}(\mathcal{B})^1$ and we prove that

$$m \cdot \underset{v}{(\!|}\, \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \,\underset{v}{|\!)}^{\tau} \quad \leqslant \quad \underset{u}{(\!|}\, \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \,\underset{u}{|\!)}^{\tau} \quad .$$

By definition of $\mathcal{B}$, $\mathcal{C}$ and $\mathcal{D}'$, every unary block in $\mathcal{B}$ is in $\mathcal{C}$ or in $\mathcal{D}'$. So, by axiom (3.2) and by Kleene algebra, it suffices to prove that

$$\underset{u}{(\!|}\, \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \,\underset{u}{|\!)}^{v} \cdot \underset{v}{(\!|}\, \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \,\underset{v}{|\!)}^{\tau} \quad \leqslant \quad \underset{u}{(\!|}\, \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \,\underset{u}{|\!)}^{\tau} \quad .$$

The proof is direct by axiom (3.4).

We now prove (6.28). We suppose that there exists a binary block $\lfloor_u c \downarrow_z \uparrow^\tau r \rfloor^\tau$ in $\mathsf{mb}(\mathcal{B}^2)$ and we prove that

$$c \cdot \underset{z}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{\tau}{|\!)} \cdot r \quad \leqslant \quad \underset{u}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{\tau}{|\!)} \ .$$

Let us do a proof by case analysis on the binary block. First, we consider the case where $\lfloor_u c \downarrow_z \uparrow^\tau r \rfloor^\tau$ is in $\mathsf{mb}(\mathcal{C})^2$. The proof is direct from the hypothesis that $\lfloor_u c \downarrow_z \uparrow^\tau r \rfloor^\tau$ is in $\mathsf{mb}(\mathcal{C})^2$ and axiom (3.3).

We now consider the case where $\lfloor_u c \downarrow_z \uparrow^\tau r \rfloor^\tau$ is in $\mathsf{mb}(\mathcal{D})^2$. We took the associated binary block $\lfloor_u c \downarrow_w \uparrow^\tau r \rfloor^\tau$ in $\mathsf{mb}(\mathcal{D}')^2$. There are two possible cases: either $w = z$, or $z = f$ and $w = f'$. In the case that $w = z$, the proof is direct from the hypothesis that $\lfloor_u c \downarrow_w \uparrow^\tau r \rfloor^\tau$ is in $\mathsf{mb}(\mathcal{D}')^2$ and axiom (3.3). For the case that $z = f$ and $w = f'$, by (6.25), it suffices to prove that

$$c \cdot \underset{f'}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{\tau}{|\!)} \cdot r \quad \leqslant \quad \underset{u}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{\tau}{|\!)} \ .$$

The proof is direct from the hypothesis that $\lfloor_u c \downarrow_{f'} \uparrow^\tau r \rfloor^\tau$ is in $\mathsf{mb}(\mathcal{D}')^2$ and axiom (3.3).

We now prove (6.29). We suppose that there exists a binary block $\lfloor_u c \downarrow_z \uparrow^\tau r \rfloor^v$ in $\mathsf{mb}(\mathcal{B})^2$ and we prove that

$$c \cdot \underset{z}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{\tau}{|\!)} \cdot r \cdot \underset{v}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{\tau}{|\!)} \quad \leqslant \quad \underset{u}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{\tau}{|\!)} \ .$$

Let us do a proof by case analysis on the binary block. First, we consider the case where $\lfloor_u c \downarrow_z \uparrow^\tau r \rfloor^v$ is in $\mathsf{mb}(\mathcal{C})^2$. By the hypothesis that $\lfloor_u c \downarrow_z \uparrow^\tau r \rfloor^v$ is in $\mathsf{mb}(\mathcal{C})^2$, axiom (3.3) and by Kleene algebra, it suffices to prove that

$$\underset{u}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{v}{|\!)} \cdot \underset{v}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{\tau}{|\!)} \quad \leqslant \quad \underset{u}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{\tau}{|\!)} \ .$$

The proof is direct by axiom (3.4).

We now consider the case where $\lfloor_u c \downarrow_z \uparrow^\tau r \rfloor^v$ is in $\mathsf{mb}(\mathcal{D})^2$. We took the associated binary block $\lfloor_u c \downarrow_w \uparrow^\tau r \rfloor^v$ in $\mathsf{mb}(\mathcal{D}')^2$. There are two possible cases: either $w = z$, or $z = f$ and $w = f'$. In the case that $w = z$, by the hypothesis that $\lfloor_u c \downarrow_w \uparrow^\tau r \rfloor^v$ is in $\mathsf{mb}(\mathcal{D}')^2$, by axiom (3.3) and by Kleene algebra, it suffices to prove that

$$\underset{u}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{v}{|\!)} \cdot \underset{v}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{\tau}{|\!)} \quad \leqslant \quad \underset{u}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{\tau}{|\!)} \ .$$

```
int main(void) {
  int x = 2;

  x += 1;
  x += 1;

  return 0; /* Success. */
}
```

Figure 6.12: Optimized version of the simple non-recursive program example of Figure 6.9a.

The proof is direct by axiom (3.4).

For the case that $z = f$ and $w = f'$, by (6.25), it suffices to prove that

$$c \cdot \underset{f'}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{\tau}{|\!)} \cdot r \cdot \underset{v}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{\tau}{|\!)} \quad \leqslant \quad \underset{u}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{\tau}{|\!)} \ .$$

By the hypothesis that $[\!]_u \, c{\downarrow_{f'}} {\uparrow^{\tau}} \, r \, [\!]^v$ is in $\mathsf{mb}(\mathcal{D}')^2$, by axiom (3.3) and by Kleene algebra, it suffices to prove that

$$\underset{u}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{v}{|\!)} \cdot \underset{v}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{\tau}{|\!)} \quad \leqslant \quad \underset{u}{(\!|} \mathsf{mb}(\mathcal{D}', \mathcal{C}, \mathcal{C}') \overset{\tau}{|\!)} \ .$$

The proof is direct by axiom (3.4). ■

### 6.3.6 Linking The Analyses Together

Each verification of an interprocedural compiler optimization that is presented in this section is done only for a specified optimizing transformation. However, our reasoning can be used through a more extensive verification (involving several optimizing transformations) to make sure the optimized program still behaves the same as the unoptimized program. It allows us, for example, to verify that the program of Figure 6.9a can be optimized to the program of Figure 6.12. This is done by using the verifications of Section 6.3.1 and Section 6.3.2, and the intraprocedural verification of copy propagation of [33].

## 6.4 Discussion

There already exist variants of Kleene algebra for the formal verification of programs with or without procedures [7, 18, 21, 30, 39, 44, 50]. However, all these frameworks only allow a user to verify *regular* properties, whereas we are able to deal with some *non-regular* properties. Of these frameworks, the only one that can deal with (mutually) recursive programs is the work of Mathieu and Desharnais [39]. It uses pushdown systems as programs and uses an extension of omega algebra with domain (that adds laws to represent explicit stack manipulations) along with matrices on this algebra to represent programs. We do not need such mechanisms.

De Moor et al. did an interesting work at the border of Kleene algebra and model checking [48]. They represented programs by context-free languages, properties by regular languages and they used residuals to isolate program parts. It is an interesting technique, but De Moor et al. are still limited to regular properties.

The model checking community already works on the formal verification of interprocedural programs [1, 12, 46]. However, most of the tools developed so far can just deal with regular properties except, of course, tools using visibly pushdown automata (that are able to deal with some non-regular properties). Note that our work is different because

(i) the verification process looks for the existence of a proof in the algebraic system,

(ii) we can encode proofs *à la* Proof-Carrying Code, and

(iii) we are not limited to decidable problems when using the proof method. The unrestricted use of hypotheses allows us to state undecidable problems. It is not clear whether these hypotheses can be used to reach a precision in the program data flow that is not possible using only visibly pushdown automata.

In the process of verifying several interprocedural optimizations in VPKAT, we sometimes had to rely on a compiler's trusted functionality to generate equational hypotheses apart from the standard hypotheses coming from the semantics of the programming language. Such dependence on some compiler's trusted functionalities may seem a flaw at first, but it appears to us more like a winning situation: VPKAT does not need to reinvent the wheel; VPKAT may benefit from already trusted analyzers that can merge well in our verification process. Verifications in VPKAT are done only in situations where it is useful (where other analyzers do not succeed). Note that the verification of compiler optimizations in pure Kleene algebra has the same dependence [33].

One may ask why VPKAT must rely on equational hypotheses to encode inlining assumptions or tail-recursion elimination. The reason is linked with the notion of equality in VPKAT: it characterizes the equality of the languages denoted by two visibly pushdown regular expressions. In other words, equality represents *trace equivalence* and, in this equality, each call action and return action is important. Those actions cannot be "deleted" or "forgotten" from the expression. So, equational hypotheses are necessary in VPKAT to extend the notion of equality around specified call actions and return actions.

Also, one may ask if some of our reasoning in this chapter may lead to the *dead variable paradox* of Kozen and Patron [33]. In particular, for the reasoning of Section 6.3.1, the test a seems a dangerous one since it is linked to the existence of a variable. However, we do not have such a paradox here. The test that checks whether the address of the variable x exists is a *property of the local state of the computation* unlike the proposition "x is a dead variable". In particular, the test a commutes with any test involving x that is a property of the local state of the computation because its existence cannot be changed by such a test.

# Chapter 7

# Conclusion

This dissertation developed an algebraic formalism that I call *visibly pushdown Kleene algebra* (VPKA), and provided evidence that the formalism can be used in some interprocedural program analyses and that these analyses could be automated. I briefly review all these aspects in Section 7.1. Then, I discuss of the open problems and future work in Section 7.2.

## 7.1 Review of the Evidence Provided to Support the Thesis

**VPKA is a Kleene-like algebraic formalism:** I first presented, in Chapter 3, the proposed formalism: visibly pushdown Kleene algebra. The formalism is based on state transitions (imperative style of programming) via an analogy to visibly pushdown automata. In fact, VPKA is formed by adding a family of implicit least fixed point operators, based on a restriction of context-free grammars, to the standard operators of Kleene algebra (which already represents equality of regular languages). So, the proposed formalism is linked to two well-known notions in computer science: finite automata and context-free grammars. The generated proofs in VPKA are equational and machine-verifiable. Moreover, the mathematical style used by the algebra is close to the mathematical style that is learned in high school.

VPKA has been developed to extend the metaphor introduced by Kleene algebra with tests concerning imperative programming languages. Kleene algebra with tests separates the notion of tests and instructions of a program in the syntax of the algebra.

Also, the operators of the algebra are directly linked with well-known constructs of imperative programming languages: sequences, alternatives and loops. I extended the metaphor to encompass the notion of code blocks (or procedures or functions) via the family of operators introduced in VPKA and the notion of metablocks (presented in Chapter 5) to further enhance the similarity. Thus, the usual basic constructs of imperative programming languages are linked to operators of the proposed formalism.

**VPKA can be used in some interprocedural program analyses:** I showed that visibly pushdown Kleene algebra extended with tests is suitable to do some interprocedural program analyses in Chapter 6. In particular, I sketched a framework to do formal verification of interprocedural programs. The framework is similar to [7] or [30], but procedures are added to the definition of programs and the verification of non-regular properties is possible. I also sketched a framework to do certification of several interprocedural compiler optimizations including interprocedural dead code elimination and inlining of functions. Thus, visibly pushdown Kleene algebra is a versatile formalism. It is possible to use, within a compiler, the same formalism to do the formal verification of a program and the certification of its optimization.

**Analyses done using VPKA could be automated:** Interprocedural program analyses done in VPKA reduce to extracting a model of a program and proving that a formula is a theorem of visibly pushdown Kleene algebra extended with tests. Of course, extracting a model of a program that is sufficient for a given verification is a hard problem. However, some heuristics already exist that give interesting approximations in practice. For this part, it is possible to reuse work done, for example, for Boolean programs [3]. So, this part could be automated if we accept that it also sometimes fails.

For the second part, we need a way to prove that a formula is a theorem of visibly pushdown Kleene algebra extended with tests. There is great hope that an interesting subset of the theory of VPKAT could be automated. In fact, I proved in Chapter 4 that the algebraic system of VPKA is sound and complete for the equational theory of visibly pushdown languages and I also showed that its complexity is EXPTIME-complete. The described algorithm is linked with the theory of visibly pushdown automata and does not directly give a proof of a theorem (it just verifies that the theorem holds), but it is an important step to define such an algorithm. There should exist an algorithm that also gives the proof of the theorem within the same boundaries. Fortunately, Kleene algebra can inspire us in this direction. Note that Kleene algebra had the same "problem" until recently. Worthington defined in 2008 a proof system for the equational theory of Kleene algebra that produces a proof of a theorem by using a PSPACE transducer [52, 53]. It should be possible to do something similar in VPKA. On the other hand, we can also simply use a generic theorem prover and try to use it for our verification purposes.

## 7.2 Open Problems and Future Work

The axiomatization of visibly pushdown Kleene algebra is particular. For example, axioms (3.14) to (3.17) are committed to the "direction" of the derivation strategy. This possibility to use a forward or a backward strategy is important when proving theorems about a program since one of the strategies is usually easier to use than the other (depending on the program). On the other hand, the $(\!|\,|\!)$-simulation axioms (axioms (3.16) and (3.17)) of VPKA seem too complex. We would like to replace them by simpler axioms. Fortunately, there is hope in this area because the completeness proof (the proof of Theorem 4.2) uses the axiom (3.17) only *once* (in the determinization step) and the axiom (3.16) only *twice* (in the determinization step and in the synchronization step). So, these axioms are not used often during the proof. It makes sense to think of replacing them with other axioms. For the moment, we have not found an interesting replacement for these axioms and we still do not know whether they are really independent from the other axioms.

For future work, one idea is to extend VPKA with "infinite" operators to represent visibly pushdown $\omega$-languages as defined in [1]. This would allow us to handle non-halting programs. Also, residuals can be added to the algebra, since visibly pushdown languages are prefix-closed. Moreover, it seems interesting to add a domain operator since it can give an algebra related to the extension of PDL defined in [38]. More than that, other applications for VPKA need to be investigated, like the verification of the correct transformation of a recursive algorithm into an iterative algorithm, or the representation of programs having nonlocal transfer of control (for example, **goto** statements). Kozen already did the latter for modular programs [32], but the proof is very involved. We think that using continuations (the ending label of a block acts like a continuation) would help to reduce the complexity of the representation and would be easier to read. The representation of programs having an exception handling mechanism in VPKA is also a desired goal to achieve.

# Bibliography


[1] R. Alur and P. Madhusudan. Visibly pushdown languages. In *STOC '04: Proc. of the 36th ACM symp. on Theory of computing*, pages 202–211, New York, USA, 2004. ACM.

[2] R. Alur and P. Madhusudan. Adding nesting structure to words. *Journal of the ACM*, 56(3):16:1–16:43, May 2009.

[3] T. Ball and S. K. Rajamani. Boolean programs: A model and process for software analysis. Technical Report MSR-TR-2000-14, Microsoft Research, 2000.

[4] K. Beck. *Test Driven Development: By Example.* Addison Wesley, 2003.

[5] S. L. Bloom and Z. Ésik. *Iteration theories: the equational logic of iterative processes.* Springer, New York, USA, 1993.

[6] C. Bolduc. Oméga-algèbre: Théorie et application en vérification de programmes. Master's thesis, Université Laval, 2006. In French.

[7] C. Bolduc and J. Desharnais. Static analysis of programs using omega algebra with tests. In *Relational Methods in Computer Science 2005*, volume 3929 of *Lecture Notes in Computer Science*, pages 60–72. Springer, 2006.

[8] C. Bolduc and B. Ktari. Visibly pushdown Kleene algebra. Technical Report DIUL-RR-0901, Université Laval, QC, Canada, 2009.

[9] C. Bolduc and B. Ktari. Visibly pushdown Kleene algebra and its use in interprocedural analysis of (mutually) recursive programs. In *Relations and Kleene Algebra in Computer Science*, volume 5827 of *Lecture Notes in Computer Science*, pages 44–58. Springer, 2009.

[10] C. Bolduc and B. Ktari. Verification of common interprocedural compiler optimizations using visibly pushdown Kleene algebra. In *AMAST 2010*, volume 6486 of *Lecture Notes in Computer Science*, pages 28–43. Springer, 2010.

[11] C. Bolduc and B. Ktari. Verification of common interprocedural compiler optimizations using visibly pushdown Kleene algebra (extended version). Technical Report DIUL-RR-1001, Université Laval, QC, Canada, 2010.

[12] A. Bouajjani, J. Esparza, and O. Maler. Reachability analysis of pushdown automata: Application to model-checking. In *Proc. of the 8th Int. Conf. on Concurrency Theory (CONCUR '97)*, volume 1243 of *Lecture Notes in Computer Science*, pages 135–150. Springer, 1997.

[13] S. Chaudhuri and R. Alur. Instrumenting C programs with nested word monitors. In *SPIN*, volume 4595 of *Lecture Notes in Computer Science*, pages 279–283, 2007.

[14] P. Chervet and I. Walukiewicz. Minimizing variants of visibly pushdown automata. In *Mathematical Foundations of Computer Science 2007*, volume 4708 of *Lecture Notes in Computer Science*, pages 135–146. Springer, 2007.

[15] E. Cohen. Separation and reduction. In *Proc. of the 5th Int. Conf. on Mathematics of Program Construction (MPC'00)*, volume 1837 of *Lecture Notes in Computer Science*, pages 45–59. Springer, 2000.

[16] J. H. Conway. *Regular Algebra and Finite Machines*. Chapman and Hall, 1971.

[17] P. Cousot and R. Cousot. Abstract interpretation and application to logic programs. *Journal of Logic Programming*, 13(2–3):103–179, 1992.

[18] J. Desharnais, B. Möller, and G. Struth. Modal Kleene algebra and applications—a survey. *Journal on Relational Methods in Computer Science*, 1:93–131, 2004.

[19] O. Grumberg and H. Veith, editors. *25 Years of Model Checking—History, Achievements, Perspectives*, volume 5000 of *Lecture Notes in Computer Science*. Springer, 2008.

[20] C. Hardin and D. Kozen. On the elimination of hypotheses in Kleene algebra with tests. Technical Report 2002-1879, Computer Science Department, Cornell University, Oct. 2002.

[21] C. A. R. Hoare, B. Möller, G. Struth, and I. Wehrman. Concurrent Kleene algebra. In *CONCUR 2009*, volume 5710 of *Lecture Notes in Computer Science*, pages 399–414. Springer, 2009.

[22] H. Hosoya, J. Vouillon, and B. C. Pierce. Regular expression types for XML. *ACM Transactions on Programming Languages and Systems*, 27(1):46–90, Jan. 2005.

[23] D. M. Kaplan. Regular expressions and the equivalence of programs. *Journal of Computer and System Sciences*, 3(4):361–386, 1969.

[24] S. C. Kleene. Representation of events in nerve nets and finite automata. In C. Shannon and J. McCarthy, editors, *Automata Studies*, pages 3–41. Princeton University Press, Princeton, NJ, 1956.

[25] D. Kozen. A completeness theorem for Kleene algebras and the algebra of regular events. *Information and Computation*, 110(2):366–390, May 1994.

[26] D. Kozen. Kleene algebra with tests and commutativity conditions. Technical Report 1996-1563, Computer Science Department, Cornell University, 1996.

[27] D. Kozen. Kleene algebra with tests. *Transactions on Programming Languages and Systems*, 19(3):427–443, May 1997.

[28] D. Kozen. On Hoare logic and Kleene algebra with tests. *ACM Transactions on Computational Logic (TOCL)*, 1(1):60–76, 2000.

[29] D. Kozen. Automata on guarded strings and applications. *Matématica Contemporânea*, 24:117–139, 2003.

[30] D. Kozen. Kleene algebras with tests and the static analysis of programs. Technical Report TR2003-1915, Cornell University, NY, USA, Nov. 2003.

[31] D. Kozen. *Introduction to Kleene Algebra.* Lecture notes. Computer Science Department, Cornell University, Spring 2004.

[32] D. Kozen. Nonlocal flow of control and Kleene algebra with tests. In *Proc. 23rd IEEE Symp. Logic in Computer Science (LICS'08)*, pages 105–117, June 2008.

[33] D. Kozen and M.-C. Patron. Certification of compiler optimizations using Kleene algebra with tests. In *Proc. 1st Int. Conf. Computational Logic (CL2000)*, volume 1861 of *Lecture Notes in Artificial Intelligence*, pages 568–582. Springer-Verlag, July 2000.

[34] D. Kozen and F. Smith. Kleene algebra with tests: Completeness and decidability. In D. van Dalen and M. Bezem, editors, *CSL'96: Proceedings of the 10th International Worshop on Computer Science Logic*, volume 1258 of *Lecture Notes in Computer Science*, pages 244–259, 1996.

[35] W. Kuich. Semirings and formal power series: Their relevance to formal languages and automata. *Handbook of formal languages, vol. 1: Word, language, grammar*, pages 609–677, 1997.

[36] H. Leiß. Towards Kleene algebra with recursion. In *Proc. of the 5th Workshop on Computer Science Logic (CSL '91)*, volume 626 of *Lecture Notes in Computer Science*, pages 242–256. Springer, 1992.

[37] H. Leiß. Kleene modules and linear languages. *Journal of Logic and Algebraic Programming*, 66(2):185–194, 2006.

[38] C. Löding, C. Lutz, and O. Serre. Propositional dynamic logic with recursive programs. *Journal of Logic and Algebraic Programming*, 73:51–69, 2007.

[39] V. Mathieu and J. Desharnais. Verification of pushdown systems using omega algebra with domain. In *Relational Methods in Computer Science 2005*, volume 3929 of *Lecture Notes in Computer Science*, pages 188–199. Springer, 2006.

[40] K. Mehlhorn. Pebbling mountain ranges and its application to DCFL-recognition. In *Automata, languages and programming (ICALP-80): 7th annual international colloquium*, volume 85 of *Lecture Notes in Computer Science*, pages 422–435. Springer, 1980.

[41] B. Möller and G. Struth. Algebras of modal operators and partial correctness. *Theoretical Computer Science*, 351(2):221–239, 2006.

[42] D. North. Behavior modification. *Better software magazine*, 8(3), Mar. 2006.

[43] C. Pitcher. Visibly pushdown expression effects for XML stream processing. In *Programming Language Technologies for XML*, pages 1–14, 2005.

[44] R. Qiao, J. Wu, and X. Gao. Probabilistic modal Kleene algebra and Hoare-style logic. In *Proceedings of the 2008 Fourth International Conference on Natural Computation*, pages 652–661. IEEE Computer Society, 2008.

[45] J. A. Robinson and A. Voronkov, editors. *Handbook of Automated Reasoning (in 2 volumes)*. Elsevier and MIT Press, 2001.

[46] S. Schwoon. *Model-Checking Pushdown Systems*. PhD thesis, Technische Universität München, 2002.

[47] A. Selman. Completeness of calculii for axiomatically defined classes of algebras. *Algebra Universalis*, 2(1):20–32, 1972.

[48] G. Sittampalam, O. de Moor, and K. F. Larsen. Incremental execution of transformation specifications. In *Proc. of the 31st ACM symp. on Principles of Programming Languages (POPL '04)*, pages 26–38, New York, USA, 2004. ACM.

[49] B. von Braunmühl and R. Verbeek. Input driven languages are recognized in $\log n$ space. In *Selected papers of the 1983 int. conf. on "Foundations of Computation Theory" on Topics in the theory of computation*, pages 1–19, New York, NY, USA, 1985. Elsevier North-Holland, Inc.

[50] J. von Wright. From Kleene algebra to refinement algebra. In *Proc. of the 6th Int. Conf. on Mathematics of Program Construction (MPC'02)*, volume 2386 of *Lecture Notes in Computer Science*, pages 233–262. Springer, 2002.

[51] K. Wagner. Eine Axiomatisierung der Theorie der regulären Folgenmengen. *Elektronische Informationsverarbeitung und Kybernetik (EIK)*, 12:337–354, 1976. In German.

[52] J. M. Worthington. Automatic proof generation in Kleene algebra. In *Relations and Kleene Algebra in Computer Science*, volume 4988 of *Lecture Notes in Computer Science*, pages 382–396. Springer, 2008.

[53] J. M. Worthington. *Automata, representations, and proofs*. PhD thesis, Cornell University, 2009.

# Appendix A

# Proof that Well-Matched Languages Generated by WMVPGs and VPGs Coincide (Theorem 3.8)

We prove that WMVPGs generate the same class of languages as VPGs restricted to well-matched VPLs. The proof of this result is eased by an intermediary class of grammars, called "WMVPG restricted for ending rules", that generate the same class of languages as VPGs restricted to well-matched VPLs and WMVPGs.

**Definition A.1** (WMVPG restricted for ending rules)**.** Let $\Sigma_\mathbf{i}$, $\Sigma_\mathbf{c}$ and $\Sigma_\mathbf{r}$ be disjoint finite sets of atomic elements. Let $V$ be a finite set of symbols containing symbol $S$. Let $\tau$ be another symbol such that $\tau \notin V$. Let $N(V \cup \{\tau\}) := \{P_{(x,y)} \mid x, y \in V \cup \{\tau\}\}$. A WMVPG *restricted for ending rules* over $\Sigma_\mathbf{i}$, $\Sigma_\mathbf{c}$ and $\Sigma_\mathbf{r}$ is a tuple $G := (V \cup \{\tau\}, P_{(S,\tau)}, \rightarrow)$ where $N(V \cup \{\tau\})$ is the set of nonterminals, $P_{(S,\tau)}$ is the starting nonterminal and $\rightarrow$ is a finite set of *explicit rewrite rules* of the form

- $P_{(X,\tau)} \rightarrow \varepsilon$ where $X \in V$;
- $P_{(X,Y)} \rightarrow a$ where $a \in \Sigma_\mathbf{i}$ and $X, Y \in V$;
- $P_{(X,Y)} \rightarrow c\, P_{(Z,\tau)}\, r$ where $c \in \Sigma_\mathbf{c}$, $r \in \Sigma_\mathbf{r}$ and $X, Y, Z \in V$

and *implicit rewrite rules* of the form

- $P_{(X,Y)} \rightarrow P_{(X,Z)}\, P_{(Z,Y)}$ for each $X, Y, Z \in V \cup \{\tau\}$. ♦

Intuitively, WMVPGs restricted for ending rules are WMVPGs such that $\varepsilon$-rules are restricted to nonterminals of the form $P_{(X,\tau)}$ and each nonterminal $P_{(X,Y)}$ that begins a height of a well-matched subword is such that $Y = \tau$.

We first prove in Section A.1 that VPGs restricted to well-matched VPLs generate the same class of languages as WMVPGs restricted for ending rules. Then, in Section A.2, we prove that WMVPGs restricted for ending rules generate the same class of languages as WMVPGs.

## A.1 Proof that VPGs Restricted to Well-Matched VPLs Generate the Same Set of Languages as WMVPGs Restricted for Ending Rules

First, note that there is a one-to-one correspondence between VPGs restricted to well-matched VPLs and the following grammars.

**Definition A.2** (WMVPG restricted for ending rules using a forward strategy)**.** Let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be disjoint finite sets of atomic elements. Let $V$ be a finite set of symbols containing symbol $S$. Let $\tau$ be another symbol such that $\tau \notin V$. Let $N(V \cup \{\tau\}) := \{P_{(x,y)} \mid x, y \in V \cup \{\tau\}\}$. A WMVPG *restricted for ending rules using a forward strategy* over $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ is a tuple $G := (V \cup \{\tau\}, P_{(S,\tau)}, \to)$ where $N(V \cup \{\tau\})$ is the set of nonterminals, $P_{(S,\tau)}$ is the starting nonterminal and $\to$ is a finite set of *explicit rewrite rules* of the form

1. $P_{(X,\tau)} \to \varepsilon$ where $X \in V$;
2. $P_{(X,Y)} \to a$ where $a \in \Sigma_{\mathbf{i}}$ and $X, Y \in V$;
3. $P_{(X,Y)} \to c\, P_{(Z,\tau)}\, r$ where $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $X, Y, Z \in V$

and *implicit rewrite rules* of the form $P_{(X,\tau)} \to P_{(X,Y)}\, P_{(Y,\tau)}$ for each $X, Y \in V$ such that the nonterminal $P_{(X,Y)}$ is already used in the left-hand side of an explicit rewrite rule of the form 2 or of the form 3. ⧫

Here is a quick proof showing the one-to-one correspondence between VPGs restricted to well-matched VPLs and WMVPGs restricted for ending rules using a forward strategy.

*Proof.* Note that for WMVPGs restricted for ending rules using a forward strategy, every nonterminal $P_{(X,Y)}$ generated by an implicit rewrite rule $P_{(X,\tau)} \to P_{(X,Y)}\, P_{(Y,\tau)}$ can be derived only by explicit rewrite rules of the form 2 or of the form 3. By the previous remark and since WMVPGs restricted for ending rules using a forward strategy all have a starting nonterminal $P_{(S,\tau)}$, then WMVPGs restricted for ending rules using a forward strategy are equivalent to grammars $(\{P_{(X,\tau)} \mid X \in V\}, P_{(S,\tau)}, \to)$ where $\{P_{(X,\tau)} \mid X \in V\}$ is the set of nonterminals, $P_{(S,\tau)}$ is the starting nonterminal and $\to$ is a finite set of rewrite rules of the form

- $P_{(X,\tau)} \to \varepsilon$ where $X \in V$;
- $P_{(X,\tau)} \to a\, P_{(Y,\tau)}$ where $a \in \Sigma_{\mathbf{i}}$ and $X, Y \in V$;
- $P_{(X,\tau)} \to c\, P_{(Z,\tau)}\, r\, P_{(Y,\tau)}$ where $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $X, Y, Z \in V$.

These grammars are easily seen to be equivalent to VPGs restricted to well-matched VPLs by replacing each $P_{(X,\tau)}$ by $X$ for any $X \in V$. ■

The difference between WMVPGs restricted for ending rules and WMVPGs restricted for ending rules using a forward strategy is the implicit rewrite rules: there is more implicit rewrite rules in WMVPGs restricted for ending rules. Note that every WMVPG restricted for ending rules using a forward strategy is associated uniquely to a WMVPG restricted for ending rules and vice versa. Thus, every word generated by a WMVPG restricted for ending rules using a forward strategy is also generated by its associated WMVPG restricted for ending rules. So, to prove that VPGs restricted to well-matched VPLs generate the same class of languages as WMVPGs restricted for ending rules, it suffices to prove that the additional implicit rewrite rules in WMVPGs restricted for ending rules do not allow the grammar to generate new words.

*Proof.* We show that for all $n \in \mathbb{N}$ and $X \in V$, every well-matched word $w$ of height $n$ generated by the rewrite rules of a WMVPG restricted for ending rules $G$ when starting in $P_{(X,\tau)}$ can also be generated by the rewrite rules of its associated WMVPG restricted for ending rules using a forward strategy $G'$ when starting in $P_{(X,\tau)}$. We prove this by generalized induction over $n$.

For the base case $n = 0$, the well-matched word $w$ is just a sequence of internal actions. Take any derivation tree that generates $w$ using the rewrite rules of $G$ and starting with $P_{(X,\tau)}$. By Definition A.1, the parent node of every leaf of the tree (here, the leafs are internal actions or the empty word) comes from the explicit rewrite rules

$P_{(U,\tau)} \to \varepsilon$ or $P_{(U,U')} \to a$ for $a \in \Sigma_{\mathbf{i}}$ and $U, U' \in V$. Moreover, by Definition A.1, it is easy to see that, when reading the tree left-to-right, the sequence of nonterminals of the explicit rewrite rules follow the pattern:

$$\sigma := P_{(U_1,U_2)}\, P_{(U_2,U_3)}\, P_{(U_3,U_4)} \cdots P_{(U_{k-1},U_k)}\, P_{(U_k,\tau)} \ ,$$

in which $U_i \in V$ for all $i \in \{1, 2, \ldots, k\}$, and, of course, $U_1 = X$. Note that, by Definition A.1, each nonterminal $P_{(U_i,U_{i+1})}$ (where $1 \leqslant i \leqslant k-1$) of $\sigma$ is rewritten using a rule of the form $P_{(U_i,U_{i+1})} \to a$ for $a \in \Sigma_{\mathbf{i}}$. Using the sequence $\sigma$, we can construct another valid derivation tree that uses only the rewrite rules of $G'$. For the case where there is only one nonterminal in the sequence, this is trivial. For the other cases, we construct the tree from bottom to top. We start with the last two nonterminals ($P_{(U_{k-1},U_k)}$ and $P_{(U_k,\tau)}$). Note that the rule $P_{(U_{k-1},\tau)} \to P_{(U_{k-1},U_k)}\, P_{(U_k,\tau)}$ exists since $P_{(U_{k-1},U_k)}$ is a nonterminal coming from a rewrite rule of the form $P_{(U_{k-1},U_k)} \to a$ for an $a \in \Sigma_{\mathbf{i}}$ by hypothesis. Then, we consider the nonterminal $P_{(U_{k-2},U_{k-1})}$. By hypothesis, this nonterminal comes from a rewrite rule of the form $P_{(U_{k-2},U_{k-1})} \to a$ for an $a \in \Sigma_{\mathbf{i}}$. So, by Definition A.2, the rewrite rule $P_{(U_{k-2},\tau)} \to P_{(U_{k-2},U_{k-1})}\, P_{(U_{k-1},\tau)}$ exists. This idea can be reused for every $P_{(U_i,U_{i+1})}$ in the sequence by reading them right-to-left. Hence, we have created a valid derivation tree for the word $w$ using the rewrite rules of $G'$ and starting with $P_{(U_1,\tau)}$ (and considering that $U_1 = X$).

For the inductive case, we suppose that for all $j \in \{0, 1, \ldots, l\}$ and $X \in V$, every well-matched word of height $j$ generated by the rewrite rules of a WMVPG restricted for ending rules $G$ when starting in $P_{(X,\tau)}$ can also be generated by the rewrite rules of its associated WMVPG restricted for ending rules using a forward strategy $G'$ when starting in $P_{(X,\tau)}$. We prove that for all $X \in V$, every well-matched word $w$ of height $l+1$ generated by the rewrite rules of $G$ when starting in $P_{(X,\tau)}$ can also be generated by the rewrite rules of $G'$ when starting in $P_{(X,\tau)}$.

Take any derivation tree that generates $w$ using the rewrite rules of $G$ and starting with $P_{(X,\tau)}$. By Definition A.1, the parent node of every leaf of the tree comes from the explicit rewrite rules of the grammar. Moreover, by Definition A.1, it is easy to see that, when reading left-to-right the nonterminals of any leaf that is not a return action (so it is either an internal action, a call action or the empty word), and omitting the nonterminals that have as ancestor a nonterminal coming from an explicit rewrite rule of the form $P_{(U,U')} \to c\, P_{(Z,\tau)}\, r$, the sequence of nonterminals of the explicit rewrite rules follow the pattern:

$$\sigma := P_{(U_1,U_2)}\, P_{(U_2,U_3)}\, P_{(U_3,U_4)} \cdots P_{(U_{k-1},U_k)}\, P_{(U_k,\tau)} \ ,$$

in which $U_i \in V$ for all $i \in \{1, 2, \ldots, k\}$, and, of course, $U_1 = X$. Note that the nonterminal of a rewrite rule of the form $P_{(U,U')} \to c\, P_{(Z,\tau)}\, r$ appears once (not twice)

in the sequence for a pair of elements $c$ and $r$. Using this sequence, we can construct another valid derivation tree that uses only the rewrite rules of $G'$. We first use the induction hypothesis. For any nonterminal $P_{(U_i,U_{i+1})}$ in the sequence that is linked to a rule of the form $P_{(U,U')} \to c\, P_{(Z,\tau)}\, r$, the induction hypothesis says that we can find a valid derivation tree for $P_{(Z,\tau)}$ since the height of the well-matched subword generated by $P_{(Z,\tau)}$ is smaller or equals to $l$. Then, we just use a reasoning similar to the base case and this allows us to generate a valid derivation tree for $w$ using the rewrite rules of $G'$ and starting with $P_{(U_1,\tau)}$ (and considering that $U_1 = X$). ■

## A.2 Proof that WMVPGs Restricted for Ending Rules Generate the Same Set of Languages as WMVPGs

We now show that WMVPGs generate the same class of languages as the WMVPGs restricted for ending rules. Since every WMVPG restricted for ending rules is also a WMVPG, it suffices to prove that for every WMVPG $G$, we can find a WMVPG restricted for ending rules $G'$ that accepts the same language.

*Proof.* The proof is done by several transformation steps. Some transformation steps use the concept of a height-starting node. Let us call a *height-starting node* in a derivation tree the root of the tree (the nonterminal $P_{(S,T)}$) and any node encapsulating the nonterminal $P_{(Z,W)}$ of a rewrite rule $P_{(X,Y)} \to c\, P_{(Z,W)}\, r$ of a WMVPG. We now present the transformation steps.

**Step 1: Ensure that any height-starting node contains a nonterminal $P_{(X,Y)}$ such that $X \neq Y$.** We first add fresh symbols $\{X' \mid X \in V\}$ to $G$. Each symbol $X'$ will be a clone of $X$ that will ensure that any height-starting node contains a nonterminal $P_{(X,Y)}$ such that $X \neq Y$. Let $V' := V \cup \{Z' \mid Z \in V\}$ So, we take $G$ and we create a new grammar $G_2 := (V', P_{(S',T)}, \to_2)$ where the explicit rewrite rules of $\to_2$ are

1. $P_{(X,Y)} \to_2 \varepsilon \Leftrightarrow P_{(X,Y)} \to \varepsilon$, for $X, Y \in V$;
2. $P_{(X',Y)} \to_2 \varepsilon \Leftrightarrow P_{(X,Y)} \to \varepsilon$, for $X, Y \in V$;
3. $P_{(X,Y)} \to_2 a \Leftrightarrow P_{(X,Y)} \to a$, for $a \in \Sigma_\mathbf{i}$ and $X, Y \in V$;
4. $P_{(X',Y)} \to_2 a \Leftrightarrow P_{(X,Y)} \to a$, for $a \in \Sigma_\mathbf{i}$ and $X, Y \in V$;

5. $P_{(X,Y)} \to_2 c\, P_{(Z',W)}\, r \Leftrightarrow P_{(X,Y)} \to c\, P_{(Z,W)}\, r$, for $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $W, X, Y, Z \in V$;

6. $P_{(X',Y)} \to_2 c\, P_{(Z',W)}\, r \Leftrightarrow P_{(X,Y)} \to c\, P_{(Z,W)}\, r$, for $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $W, X, Y, Z \in V$.

It is easy to see that any derivation tree of $G_2$ can be converted to a valid derivation tree for $G$ and vice versa by replacing each height-starting node containing a nonterminal $P_{(X',Y)}$ (and its children generated by implicit rewrite rules of course) by the nonterminal $P_{(X,Y)}$ instead. After all, their associated rules' right-hand side are identical. Thus, the grammar $G_2$ generates the same language as $G$.

**Step 2: Add self-looping $\varepsilon$-rules to $G_2$.** For the second step, we add "self-looping" $\varepsilon$-rules to $G_2$ for all pairs of $V' \times V'$. We define a grammar $G_3 := (V', P_{(S',T)}, \to_3)$ where the explicit rewrite rules of $\to_3$ are the explicit rewrite rules of $\to_2$ along with the additional rewrite rules:

1. $P_{(X,X)} \to_3 \varepsilon$, for all $X \in V'$.

We now show that $G_3$ generates the same language as $G_2$. We first note that in any valid derivation tree for $G_3$, the parent node of any node that represents an added rewrite rule $P_{(X,X)} \to_3 \varepsilon$ is necessarily a node representing an implicit rewrite rule. After all, this is ensured by Step 1 (any height-starting node contains a nonterminal $P_{(X,Y)}$ such that $X \neq Y$). Note also that any implicit rewrite rule $P_{(X,Y)} \to_3 P_{(X,Z)}\, P_{(Z,Y)}$ such that one of the nonterminals $P_{(X,Z)}$ and $P_{(Z,Y)}$ is immediately rewritten with an added rewrite rule is such that $Z = X$ or $Z = Y$. Thus, it "yields" the derivation $P_{(X,Y)} \to_3 P_{(X,Y)}$. So, no "progress" (no new word) can be made by using the implicit rewrite rules along with the added rewrite rules. The remaining case is if the implicit rewrite rule $P_{(X,Y)} \to_3 P_{(X,Z)}\, P_{(Z,Y)}$ is such that both nonterminals $P_{(X,Z)}$ and $P_{(Z,Y)}$ are immediately rewritten with an added rewrite rule. This can happen only if $X = Y$. However, in any valid derivation tree where it can happen, the parent node of the node that represents the previous rule is necessarily a node representing an implicit rewrite rule (again, this is ensured by Step 1). So, for this parent node representing an implicit rewrite rule, the same reasoning can be applied until we find a parent node representing an implicit rewrite rule $P_{(X,Y)} \to_3 P_{(X,Z)}\, P_{(Z,Y)}$ such that $X \neq Y$. This will always happen since each height-starting node contains a nonterminal $P_{(X,Y)}$ such that $X \neq Y$.

**Step 3: Add a new starting nonterminal that can be used only once by the explicit rewrite rules.** We now add a fresh symbol $S''$ to $G_3$. This symbol will be a clone of $S'$ but it will not be possible to use it more than once with explicit rewrite rules. So, we take $G_3$ and we create a new grammar $G_4 := (V' \cup \{S''\}, P_{(S'',T)}, \rightarrow_4)$ where the explicit rewrite rules of $\rightarrow_4$ are the explicit rewrite rules of $\rightarrow_3$ along with the additional rewrite rules:

1. $P_{(S'',Y)} \rightarrow_4 \varepsilon \Leftrightarrow P_{(S',Y)} \Rightarrow_3^* \varepsilon$, for $Y \in V'$;
2. $P_{(S'',Y)} \rightarrow_4 a \Leftrightarrow P_{(S',Y)} \rightarrow_3 a$, for $a \in \Sigma_{\mathbf{i}}$ and $Y \in V'$;
3. $P_{(S'',Y)} \rightarrow_4 c\, P_{(Z,W)}\, r \Leftrightarrow P_{(S',Y)} \rightarrow_3 c\, P_{(Z,W)}\, r$, for $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $W, Y, Z \in V'$.

It is easy to see that the grammar $G_4$ generates the same language as $G_3$. After all, every rule having $P_{(S'',T)}$ on the left-hand side is identical to a rule of $G_3$ having $P_{(S',T)}$ on the left-hand side except for the rules of item 1 which represent a combination of $\varepsilon$-rules of $G_3$ and implicit rewrite rules of $G_3$.

**Step 4: Get rid of all non-self-looping explicit rewrite rules $P_{(X,Y)} \rightarrow_4 \varepsilon$ for $X, Y \in V'$.** We construct a new grammar $G_5$ that simulates an $\varepsilon$-rules closure. We define $G_5 := (V' \cup \{S''\}, P_{(S'',T)}, \rightarrow_5)$ where the explicit rewrite rules of $\rightarrow_5$ are

1. $P_{(X,X)} \rightarrow_5 \varepsilon$, for all $X \in V'$;
2. $P_{(S'',Y)} \rightarrow_5 \varepsilon \Leftrightarrow P_{(S'',Y)} \rightarrow_4 \varepsilon$, for all $Y \in V'$;
3. $P_{(X,Y)} \rightarrow_5 a \Leftrightarrow (\exists\ Z \mid Z \in V' \cup \{S''\} : P_{(X,Z)} \rightarrow_4 a\ \wedge\ P_{(Z,Y)} \Rightarrow_4^* \varepsilon)$, for all $a \in \Sigma_{\mathbf{i}}$ and $X, Y \in V' \cup \{S''\}$;
4. $P_{(X,Y)} \rightarrow_5 c\, P_{(Z,W)}\, r \Leftrightarrow (\exists\ Z_2, Y_2 \mid Z_2, Y_2 \in V' \cup \{S''\} : P_{(X,Y_2)} \rightarrow_4 c\, P_{(Z_2,W)}\, r\ \wedge\ P_{(Z_2,Z)} \Rightarrow_4^* \varepsilon\ \wedge\ P_{(Y_2,Y)} \Rightarrow_4^* \varepsilon)$, for all $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $X, Y, Z, W \in V' \cup \{S''\}$.

We now show that $G_5$ generates the same language as $G_4$. It is easy to see that if a word is generated by $G_5$, then it can be generated by $G_4$ (by saturation of the rules). We just show the other case. We show that for each valid derivation tree $t$ in $G_4$ that generates a word $w$, there exists a derivation tree in $G_5$ that also generates $w$. This derivation tree is obtained by transforming $t$. Recall that for any $X \in V'$, $P_{(X,X)} \rightarrow_4 \varepsilon$. So, all the rules of $G_4$ are in $G_5$ except for rules of the form $P_{(X,Y)} \rightarrow_4 \varepsilon$ for $X \neq Y$.

For each node of $t$ that represents a rule of the form $P_{(X,Y)} \to_4 \varepsilon$ for $X \neq Y$, start with the leftmost and find the root of its largest $\varepsilon$-tree[1] that contains the node (suppose $P_{(U,U_2)}$ is the root of this $\varepsilon$-tree). If $P_{(U,U_2)}$ is also the root of the tree, then the rule $P_{(U,U_2)} \to_5 \varepsilon$ exists since $P_{(U,U_2)} \Rightarrow_4^* \varepsilon$.

If $P_{(U,U_2)}$ is not the root of the tree and $U = S''$, then, by definition of $G_4$, $P_{(U,U_2)}$ has as parent a node representing an implicit rewrite rule of the form $P_{(S'',U_3)} \to_5 P_{(S'',U_2)}\, P_{(U_2,U_3)}$ for a $U_3 \in V'$. Thus, we have the rewrite rules $P_{(S'',U_2)} \to_5 \varepsilon$ and $P_{(S'',U_3)} \to_5 P_{(S'',U_2)}\, P_{(U_2,U_3)}$ that can be used in the derivation tree of $G_5$. After that, take the next leftmost node of the new $t$ that represents a rule of the form $P_{(X,Y)} \to_4 \varepsilon$ for $X \neq Y$, and find the root of its largest $\varepsilon$-tree (call again $P_{(U,U_2)}$ the root of this $\varepsilon$-tree, but this $U$ will be such that $U \neq S''$).

If $P_{(U,U_2)}$ is not the root of the tree and $U \neq S''$, take its parent node. We construct the derivation tree of $G_5$ by removing the root of the $\varepsilon$-tree containing the leftmost node (or the leftmost node just after the possible node representing the rewrite rule $P_{(S'',Z)} \to_5 \varepsilon$). If the parent node of $P_{(U,U_2)}$ is a node representing a rule of the form $P_{(X,Y)} \to_4 c\, P_{(U,U_2)}\, r$, then by definition of $\to_5$, there exists the rewrite rules $P_{(X,Y)} \to_5 c\, P_{(U_2,U_2)}\, r$ and $P_{(U_2,U_2)} \to_5 \varepsilon$ which are needed for the derivation tree of $G_5$.

On the other hand, if the parent node of $P_{(U,U_2)}$ represents a rule of the form $P_{(X,Y)} \to_4 P_{(X,Z)}\, P_{(Z,Y)}$, then either $P_{(U,Y)} \to_4 P_{(U,U_2)}\, P_{(U_2,Y)}$ or $P_{(X,U_2)} \to_4 P_{(X,U)}\, P_{(U,U_2)}$ but $P_{(U_2,Y)}$ and $P_{(X,U)}$ do not generate $\varepsilon$ in $t$ since by definition $P_{(U,U_2)}$ is the largest $\varepsilon$-tree that contains the required node. If the rewrite rule is $P_{(U,Y)} \to_4 P_{(U,U_2)}\, P_{(U_2,Y)}$, then it is possible to loop on the parent node of this rule until we find a rule of the form $P_{(X,Z)} \to_4 P_{(X,U)}\, P_{(U,Z)}$ or until we encounter a height-starting node.

If we encounter a height-starting node, note that it cannot be the starting nonterminal $P_{(S'',T)}$, since $U \neq S''$ by hypothesis. So, it is another height-starting node and it represents a rule of the form $P_{(X,Y)} \to_4 c\, P_{(U,W)}\, r$. By the definition of $G_5$, the rewrite rule $P_{(X,Y)} \to_5 c\, P_{(U_2,W)}\, r$ exists and can be used in the derivation tree of $G_5$. Then, we rewrite the node $P_{(U,W)}$ (and its left child, recursively, that is of the form $P_{(U,Z_3)}$ for $Z_3 \in V'$) by $P_{(U_2,W)}$ and we delete the $\varepsilon$-tree having root $P_{(U,U_2)}$.

It remains to show the case where the rewrite rule is $P_{(X,Z)} \to_4 P_{(X,U)}\, P_{(U,Z)}$. Take the tree starting at node $P_{(X,U)}$ and find its rightmost node representing an explicit rewrite rule (it is a node that we can call $P_{(Z_2,U)}$ because it must end with $U$). If $Z_2 = S''$ and the explicit rewrite rule represented by $P_{(Z_2,U)}$ is $P_{(Z_2,U)} \to_5 \varepsilon$, then it is the case that $P_{(Z_2,U_2)} \Rightarrow_4^* \varepsilon$ and so the node $P_{(Z_2,U)}$ becomes the node $P_{(Z_2,U_2)}$ that

[1] A $\varepsilon$-tree is a derivation tree that generates $\varepsilon$.

represents the rule $P_{(Z_2,U_2)} \to_5 \varepsilon$ in the derivation tree of $G_5$. Then, we rewrite the node $P_{(U,Z)}$ (and its left child, recursively, that is of the form $P_{(U,Z_3)}$ for $Z_3 \in V'$) by $P_{(U_2,Z)}$ and we delete the $\varepsilon$-tree having root $P_{(U,U_2)}$.

If $Z_2 \neq S''$ or if the explicit rewrite rule represented by $P_{(Z_2,U)}$ is not $P_{(Z_2,U)} \to_5 \varepsilon$, then we do another transformation. Since $P_{(U,U_2)}$ is the root of the $\varepsilon$-tree that contains the leftmost node representing a rewrite rule of the form $P_{(X,Y)} \to_4 \varepsilon$ for $X \neq Y$ and $X \neq S''$, then the node for $P_{(Z_2,U)}$ represents either the rewrite rule $P_{(Z_2,U)} \to_4 a$ or $P_{(Z_2,U)} \to_4 c\, P_{(W,W_2)}\, r$. By the definition of $G_5$, either the rewrite rule $P_{(Z_2,U_2)} \to_5 a$ or $P_{(Z_2,U_2)} \to_5 c\, P_{(W,W_2)}\, r$ exists and can be used in the derivation tree of $G_5$. Then, we rewrite the node $P_{(U,Z)}$ (and its left child, recursively, that is of the form $P_{(U,Z_3)}$ for $Z_3 \in V'$) by $P_{(U_2,Z)}$ and we delete the $\varepsilon$-tree having root $P_{(U,U_2)}$.

After this transformation, take the next leftmost node of the new $t$ that represents a rule of the form $P_{(X,Y)} \to_4 \varepsilon$ for $X \neq Y$ and $X \neq S''$, find the root of its largest $\varepsilon$-tree and repeat this transformation until there is no such next leftmost node.

**Step 5: Add explicit rewrite rules for the starting nonterminal that allow us to use $\varepsilon$-rules only to generate the empty word.** We define $G_6 := (V' \cup \{S''\}, P_{(S'',T)}, \to_6)$ where the explicit rewrite rules of $\to_6$ are the explicit rewrite rules of $\to_5$ along with the rules

1. $P_{(S'',Y)} \to_6 a \Leftrightarrow (\exists\ Z \mid Z \in V' \cup \{S''\} : P_{(S'',Z)} \Rightarrow_5^* \varepsilon\ \wedge\ P_{(Z,Y)} \to_5 a)$, for $a \in \Sigma_{\mathbf{i}}$ and $Y \in V' \cup \{S''\}$;
2. $P_{(S'',Y)} \to_6 c\, P_{(Z,W)}\, r \Leftrightarrow (\exists\ X \mid X \in V' \cup \{S''\} : P_{(S'',X)} \Rightarrow_5^* \varepsilon\ \wedge\ P_{(X,Y)} \to_5 c\, P_{(Z,W)}\, r)$, for $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $Y, Z, W \in V' \cup \{S''\}$.

It is easy to see that the grammar $G_6$ generates the same language as $G_5$ (simple saturation of the rules).

**Step 6: Create the** WMVPG **restricted for ending rules.** We define the grammar $G' := ((V' \cup \{S''\}) \times (V' \cup \{S''\}) \cup \{\tau\}, P_{((S'',T),\tau)}, \to_7)$ where the explicit rewrite rules of $\to_7$ are the following rules

1. $P_{((X,Y),\tau)} \to_7 \varepsilon \Leftrightarrow P_{(X,Y)} \to_6 \varepsilon$, for $X, Y \in V' \cup \{S''\}$;
2. $P_{((X,Z),(Y,Z))} \to_7 a \Leftrightarrow P_{(X,Y)} \to_6 a$, for $X, Y, Z \in V' \cup \{S''\}$;

3. $P_{((X,Z),(Y,Z))} \to_7 c\, P_{((W,W_2),\tau)}\, r \Leftrightarrow P_{(X,Y)} \to_6 c\, P_{(W,W_2)}\, r$, for $X, Y, Z, W, W_2 \in V' \cup \{S''\}$.

It is easy to see that the grammar $G'$ generates the same language as $G_6$. For the case of the word $\varepsilon$, this is trivial. For every other word, consider that every nonterminal $P_{((X,Y),\tau)}$ can always be rewritten with the implicit rewrite rule $P_{((X,Y),\tau)} \to_7 P_{((X,Y),(Y,Y))} P_{((Y,Y),\tau)}$ and the explicit rewrite rule $P_{((Y,Y),\tau)} \to_7 \varepsilon$. So, it remains to prove that every nonterminal $P_{((X,Y),(Y,Y))}$ in $G'$ can be rewritten with similar rules in $G_6$ if deriving $P_{(X,Y)}$. This is trivial by the definition of $G'$. The opposite is also true, but we need to forget intermediate $\varepsilon$-rules for all heights. In particular, step 5 allows us to not use $\varepsilon$-rules for the starting nonterminal and step 4 allows us to not use $\varepsilon$-rules in the other cases.

Thus, $G'$ generates the same language as $G$ and $G'$ is a WMVPG restricted for ending rules. ■

# Appendix B

# Proof of the Extension of Kleene's Representation Theorem for S-VPA (Theorem 4.6)

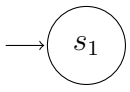
The proof of Theorem 4.6 is done by induction on the structure of $p$. Note that the base cases construct VPAs (which are a special case of S-VPAs) and follow Remark 4.4. Note also that, in our equations between matrices, we sometimes take the liberty of having different entry notation for the matrices. However, the correspondence between the notations will always be clear from the context and, of course, the matrices will always be of the same size.

## B.1 Base Case: The Constant "Zero"

A visibly pushdown automaton that accepts $\mathcal{L}(0) = \emptyset$ is

$$(\{s_1\}, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \emptyset \cup \{\bot\}, \emptyset, \{s_1\}, \emptyset) \ .$$

In other words, it is the automaton




The algebraic encoding of this automaton is

$$\left(\left[\ 1\ \right], \left[\ \left(\!\!\left(_{s_1}\ \left[_{s_1}\ 1\ \right]^{s_1}\ \right)\!\!\right)^{s_1}\ \right], \mathbf{0}, \mathbf{0}, \left[\ 0\ \right]\right).$$

We are able to prove (4.14):

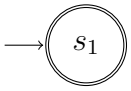
$$\begin{aligned} & \left[\ 1\ \right] \bullet \left( \mathbf{0} + \left[\ \left(\!\left|_{s_1}\right.\left[_{s_1}\ 1\ \right]^{s_1}\!\right|\!\right)^{s_1}\ \right] \right)^{\star} \bullet \left( \mathbf{0} \bullet \left[\ \left(\!\left|_{s_1}\right.\left[_{s_1}\ 1\ \right]^{s_1}\!\right|\!\right)^{s_1}\ \right] \right)^{\star} \bullet \left[\ 0\ \right] \\ = & \qquad \{\!\{ \text{ Zero of } \cdot \ \}\!\} \\ & 0 \end{aligned}$$

## B.2 Base Case: The Constant "One"

A visibly pushdown automaton that accepts $\mathcal{L}(1) = \{\varepsilon\}$ is

$$(\{s_1\}, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \emptyset \cup \{\bot\}, \emptyset, \{s_1\}, \{s_1\})\ .$$

In other words, it is the automaton




The algebraic encoding of this automaton is

$$\left( \left[\ 1\ \right], \left[\ \left(\!\left|_{s_1}\right.\left[_{s_1}\ 1\ \right]^{s_1}\!\right|\!\right)^{s_1}\ \right], \mathbf{0}, \mathbf{0}, \left[\ 1\ \right] \right).$$

We are able to prove (4.14):

$$\begin{aligned} & \left[\ 1\ \right] \bullet \left( \mathbf{0} + \left[\ \left(\!\left|_{s_1}\right.\left[_{s_1}\ 1\ \right]^{s_1}\!\right|\!\right)^{s_1}\ \right] \right)^{\star} \bullet \left( \mathbf{0} \bullet \left[\ \left(\!\left|_{s_1}\right.\left[_{s_1}\ 1\ \right]^{s_1}\!\right|\!\right)^{s_1}\ \right] \right)^{\star} \bullet \left[\ 1\ \right] \\ = & \qquad \{\!\{ \text{ Zero of } \cdot \text{ \& Kleene algebra: } 0^* = 1 \text{ \& Identity of } \cdot \text{ and } + \ \}\!\} \\ & \left[\ \left(\!\left|_{s_1}\right.\left[_{s_1}\ 1\ \right]^{s_1}\!\right|\!\right)^{s_1}\ \right]^{\star} \\ = & \qquad \{\!\{ \text{ Lemma 4.5, property iii } \}\!\} \\ & \left(\!\left|_{s_1}\right.\left[_{s_1}\ 1\ \right]^{s_1}\!\right|\!\right)^{s_1} \end{aligned}$$

It remains to show that

$$\vdash 1 = \left(\!\left|\ {}^{s_1\ s_1}_{s_1\ s_1}\left[\ 1\ \right]\ \right|\!\right) \quad . \tag{B.1}$$

The case $\leqslant$ is direct by axiom (3.2). For the case $\geqslant$,

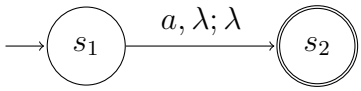
$$
\begin{array}{ll}
 & (\!|_{s_1} \lceil_{s_1} 1 \rceil^{s_1} |\!)^{s_1} \leqslant 1 \\
\leftarrow & \qquad \{\!\{ \text{ Axiom (3.14) with } s_{(s_1,s_1)} := 1 \ \}\!\} \\
 & 1 \leqslant 1 \quad \wedge \quad 1 \cdot 1 \leqslant 1 \\
 & \qquad\qquad \text{— ( Identity of } \cdot \text{ \& Reflexivity of } \leqslant \text{ . )}
\end{array}
$$

## B.3 Base Case: An Internal Action

A visibly pushdown automaton that accepts $\mathcal{L}(a) = \{a\}$ for $a \in \Sigma_{\mathbf{i}}$ is

$$(\{s_1, s_2\}, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \emptyset \cup \{\bot\}, \{(s_1, a, \lambda; s_2, \lambda)\}, \{s_1\}, \{s_2\}) \ .$$

In other words, it is the automaton




Let $\mathcal{B} := \lceil_{s_1} 1 \rceil^{s_1}, \ \lceil_{s_2} 1 \rceil^{s_2}, \ \lceil_{s_1} a \rceil^{s_2}$. The algebraic encoding of this automaton is

$$\left( \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} (\!|_{s_1} \mathcal{B} |\!)^{s_1} & (\!|_{s_1} \mathcal{B} |\!)^{s_2} \\ (\!|_{s_2} \mathcal{B} |\!)^{s_1} & (\!|_{s_2} \mathcal{B} |\!)^{s_2} \end{bmatrix}, \mathbf{0}, \mathbf{0}, \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right) .$$

We are able to prove (4.14):

$$
\begin{array}{ll}
 & \begin{bmatrix} 1 & 0 \end{bmatrix} \bullet \left( \mathbf{0} + \begin{bmatrix} (\!|_{s_1} \mathcal{B} |\!)^{s_1} & (\!|_{s_1} \mathcal{B} |\!)^{s_2} \\ (\!|_{s_2} \mathcal{B} |\!)^{s_1} & (\!|_{s_2} \mathcal{B} |\!)^{s_2} \end{bmatrix} \right)^{\star} \bullet \left( \mathbf{0} \bullet \begin{bmatrix} (\!|_{s_1} \mathcal{B} |\!)^{s_1} & (\!|_{s_1} \mathcal{B} |\!)^{s_2} \\ (\!|_{s_2} \mathcal{B} |\!)^{s_1} & (\!|_{s_2} \mathcal{B} |\!)^{s_2} \end{bmatrix} \right)^{\star} \bullet \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\
= & \qquad \{\!\{ \text{ Zero of } \cdot \text{ \& Kleene algebra: } 0^* = 1 \text{ \& Identity of } \cdot \text{ and } + \ \}\!\} \\
 & \begin{bmatrix} 1 & 0 \end{bmatrix} \bullet \begin{bmatrix} (\!|_{s_1} \mathcal{B} |\!)^{s_1} & (\!|_{s_1} \mathcal{B} |\!)^{s_2} \\ (\!|_{s_2} \mathcal{B} |\!)^{s_1} & (\!|_{s_2} \mathcal{B} |\!)^{s_2} \end{bmatrix}^{\star} \bullet \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\
= & \qquad \{\!\{ \text{ Lemma 4.5, property iii \& Definition of } \bullet \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } \\
 & \qquad\qquad + \ \}\!\} \\
 & (\!|_{s_1} \mathcal{B} |\!)^{s_2}
\end{array}
$$

It remains to show that $\vdash a = (\!|_{s_1} \mathcal{B} |\!)^{s_2}$. The case $\leqslant$ is direct by axiom (3.2). For the case $\geqslant$,

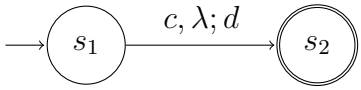
$$
\begin{array}{ll}
 & \langle\!|_{s_1}\, \mathcal{B}\, |\!\rangle^{s_2} \leqslant a \\
\leftarrow & \qquad \{\!\{ \text{ Axiom (3.14) with } s_{(s_1,s_1)} := 1,\ s_{(s_1,s_2)} := a \text{ and } s_{(s_2,s_2)} := 1 \ \}\!\} \\
 & 1 \leqslant 1 \quad \wedge \quad 1 \cdot 1 \leqslant 1 \quad \wedge \quad a \leqslant a \quad \wedge \quad 1 \cdot a \leqslant a \quad \wedge \quad a \cdot 1 \leqslant a \\
 & \qquad\qquad \text{— ( Identity of } \cdot \text{ \& Reflexivity of } \leqslant \text{ . )}
\end{array}
$$

## B.4 Base Case: A Call Action

A visibly pushdown automaton that accepts $\mathcal{L}(c) = \{c\}$ for $c \in \Sigma_{\mathbf{c}}$ is

$$(\{s_1, s_2\}, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \{d\} \cup \{\bot\}, \{(s_1, c, \lambda; s_2, d)\}, \{s_1\}, \{s_2\}) \ .$$

In other words, it is the automaton




Let $\mathcal{B} := \lfloor_{s_1}\, 1\, \rfloor^{s_1},\ \lfloor_{s_2}\, 1\, \rfloor^{s_2}$. The algebraic encoding of this automaton is

$$\left( \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} \langle\!|_{s_1}\, \mathcal{B}\, |\!\rangle^{s_1} & \langle\!|_{s_1}\, \mathcal{B}\, |\!\rangle^{s_2} \\ \langle\!|_{s_2}\, \mathcal{B}\, |\!\rangle^{s_1} & \langle\!|_{s_2}\, \mathcal{B}\, |\!\rangle^{s_2} \end{bmatrix}, \begin{bmatrix} 0 & c \\ 0 & 0 \end{bmatrix}, \mathbf{0}, \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right) .$$

First, note that

$$\vdash \begin{bmatrix} \langle\!|_{s_1}\, \mathcal{B}\, |\!\rangle^{s_1} & \langle\!|_{s_1}\, \mathcal{B}\, |\!\rangle^{s_2} \\ \langle\!|_{s_2}\, \mathcal{B}\, |\!\rangle^{s_1} & \langle\!|_{s_2}\, \mathcal{B}\, |\!\rangle^{s_2} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \ . \tag{B.2}$$

The case $\geqslant$ is direct by Lemma 4.5, property i. For the case $\leqslant$, all four inequations are proved using axiom (3.14) with $s_{(s_1,s_1)} := 1$, $s_{(s_1,s_2)} := 0$, $s_{(s_2,s_1)} := 0$ and $s_{(s_2,s_2)} := 1$. So, it suffices to prove

$$1 \leqslant 1 \quad \wedge \quad 1 \cdot 1 \leqslant 1 \quad \wedge \quad 1 \cdot 0 \leqslant 0 \ ,$$

which is trivial using identity of $\cdot$, zero of $\cdot$ and reflexivity of $\leqslant$.

We now prove (4.14).

$$\begin{bmatrix} 1 & 0 \end{bmatrix} \bullet \left( \mathbf{0} + \begin{bmatrix} \langle\!|_{s_1}\, \mathcal{B}\, |\!\rangle^{s_1} & \langle\!|_{s_1}\, \mathcal{B}\, |\!\rangle^{s_2} \\ \langle\!|_{s_2}\, \mathcal{B}\, |\!\rangle^{s_1} & \langle\!|_{s_2}\, \mathcal{B}\, |\!\rangle^{s_2} \end{bmatrix} \right)^{\star} \bullet \left( \begin{bmatrix} 0 & c \\ 0 & 0 \end{bmatrix} \bullet \begin{bmatrix} \langle\!|_{s_1}\, \mathcal{B}\, |\!\rangle^{s_1} & \langle\!|_{s_1}\, \mathcal{B}\, |\!\rangle^{s_2} \\ \langle\!|_{s_2}\, \mathcal{B}\, |\!\rangle^{s_1} & \langle\!|_{s_2}\, \mathcal{B}\, |\!\rangle^{s_2} \end{bmatrix} \right)^{\star} \bullet \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

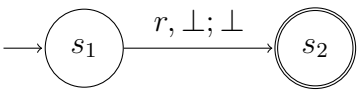
$$
\begin{aligned}
&= && \{\!\{ \text{Identity of } + \text{ \& Equation (B.2) \& Kleene algebra: } 1^* = 1 \text{ \& Identity of } \cdot \}\!\} \\
&\quad \begin{bmatrix} 1 & 0 \end{bmatrix} \bullet \begin{bmatrix} 0 & c \\ 0 & 0 \end{bmatrix}^\star \bullet \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\
&= && \{\!\{ \text{Definition of } {}^\star \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } + \}\!\} \\
&\quad \begin{bmatrix} 1 & 0 \end{bmatrix} \bullet \begin{bmatrix} 1 & c \\ 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\
&= && \{\!\{ \text{Definition of } \bullet \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } + \}\!\} \\
&\quad c
\end{aligned}
$$

## B.5 Base Case: A Return Action

A visibly pushdown automaton that accepts $\mathcal{L}(r) = \{r\}$ for $r \in \Sigma_{\mathbf{r}}$ is

$$(\{s_1, s_2\}, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \emptyset \cup \{\bot\}, \{(s_1, r, \bot; s_2, \bot)\}, \{s_1\}, \{s_2\}) \ .$$

In other words, it is the automaton




Let $\mathcal{B} := \left[_{s_1}\ 1\ \right]^{s_1},\ \left[_{s_2}\ 1\ \right]^{s_2}$. The algebraic encoding of this automaton is

$$\left( \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} (\!|_{s_1}\ \mathcal{B}\ |\!)^{s_1} & (\!|_{s_1}\ \mathcal{B}\ |\!)^{s_2} \\ (\!|_{s_2}\ \mathcal{B}\ |\!)^{s_1} & (\!|_{s_2}\ \mathcal{B}\ |\!)^{s_2} \end{bmatrix}, \mathbf{0}, \begin{bmatrix} 0 & r \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right) .$$

We are able to prove (4.14).

$$
\begin{aligned}
&\quad \begin{bmatrix} 1 & 0 \end{bmatrix} \bullet \left( \begin{bmatrix} 0 & r \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} (\!|_{s_1}\ \mathcal{B}\ |\!)^{s_1} & (\!|_{s_1}\ \mathcal{B}\ |\!)^{s_2} \\ (\!|_{s_2}\ \mathcal{B}\ |\!)^{s_1} & (\!|_{s_2}\ \mathcal{B}\ |\!)^{s_2} \end{bmatrix} \right)^\star \bullet \left( \mathbf{0} \bullet \begin{bmatrix} (\!|_{s_1}\ \mathcal{B}\ |\!)^{s_1} & (\!|_{s_1}\ \mathcal{B}\ |\!)^{s_2} \\ (\!|_{s_2}\ \mathcal{B}\ |\!)^{s_1} & (\!|_{s_2}\ \mathcal{B}\ |\!)^{s_2} \end{bmatrix} \right)^\star \bullet \\
&\quad \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\
&= && \{\!\{ \text{Zero of } \cdot \text{ \& Kleene algebra: } 0^* = 1 \text{ \& Identity of } \cdot \text{ \& Equation (B.2)} \}\!\} \\
&\quad \begin{bmatrix} 1 & 0 \end{bmatrix} \bullet \left( \begin{bmatrix} 0 & r \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right)^\star \bullet \begin{bmatrix} 0 \\ 1 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
= & \quad \{\!\{ \text{ Kleene algebra: } (p+1)^* = p^* \ \}\!\} \\
& \begin{bmatrix} 1 & 0 \end{bmatrix} \bullet \begin{bmatrix} 0 & r \\ 0 & 0 \end{bmatrix}^{\star} \bullet \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\
= & \quad \{\!\{ \text{ Definition of } {}^{\star} \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } + \ \}\!\} \\
& \begin{bmatrix} 1 & 0 \end{bmatrix} \bullet \begin{bmatrix} 1 & r \\ 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\
= & \quad \{\!\{ \text{ Definition of } \bullet \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } + \ \}\!\} \\
& r
\end{aligned}
$$

## B.6 Induction Case: Operator +

Let $p := q_1 + q_2$ where $q_1$ and $q_2$ are visibly pushdown regular expressions. Suppose there exists a semi-visibly pushdown automaton $\mathcal{A}_1 := (S_1, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_1 \cup \{\bot\}, \delta_1, I_1, F_1)$ that accepts the language $\mathcal{L}(q_1)$ and the algebraic encoding of $\mathcal{A}_1$ by the structure $(\overrightarrow{I_1}, \mathbf{WM}_1, \mathbf{T}_{\mathsf{c}_1}, \mathbf{T}_{\bot_1}, \varepsilon_{\bot_1}, \varepsilon_{\not\bot_1}, \overrightarrow{F_1})$ is such that (4.14) is valid. Also, suppose there exists a semi-visibly pushdown automaton $\mathcal{A}_2 := (S_2, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_2 \cup \{\bot\}, \delta_2, I_2, F_2)$ that accepts the language $\mathcal{L}(q_2)$ and the algebraic encoding of $\mathcal{A}_2$ by the structure $(\overrightarrow{I_2}, \mathbf{WM}_2, \mathbf{T}_{\mathsf{c}_2}, \mathbf{T}_{\bot_2}, \varepsilon_{\bot_2}, \varepsilon_{\not\bot_2}, \overrightarrow{F_2})$ is such that (4.14) is valid.

Without loss of generality, suppose $S_1 \cap S_2 = \emptyset$ and $\Gamma_1 \cap \Gamma_2 = \emptyset$. Define the semi-visibly pushdown automaton

$$\mathcal{A} := (S_1 \cup S_2, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_1 \cup \Gamma_2 \cup \{\bot\}, \delta, I_1 \cup I_2, F_1 \cup F_2)$$

where $\delta := \delta_1 \cup \delta_2 \cup \mathsf{Waste}$, and

$$
\begin{aligned}
\mathsf{Waste} := \quad & \{(s, \varepsilon, d; s', \bot) \mid d \in \Gamma_2 \wedge (\exists\ d' \mid d' \in \Gamma_1 : (s, \varepsilon, d'; s', \bot) \in \delta_1)\} \\
& \cup \{(s, \varepsilon, d; s', \bot) \mid d \in \Gamma_1 \wedge (\exists\ d' \mid d' \in \Gamma_2 : (s, \varepsilon, d'; s', \bot) \in \delta_2)\} \ .
\end{aligned}
$$

It is easy to see that $\mathcal{A}$ accepts the language $\mathcal{L}(q_1 + q_2)$. Note that the elements of $\mathsf{Waste}$ are added just to satisfy the property (2.3).

Let $\mathcal{B} := \mathcal{B}_1, \mathcal{B}_2$ (this list of block is used since $S_1 \cap S_2 = \emptyset$ and $\Gamma_1 \cap \Gamma_2 = \emptyset$ by hypothesis, and the construction just adds $\varepsilon$-transitions). The algebraic encoding of this automaton is

$$
\left( \overrightarrow{I}, \mathbf{WM}, \left[\begin{array}{c|c} \mathbf{T}_{\mathsf{c}_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}_2} \end{array}\right], \left[\begin{array}{c|c} \mathbf{T}_{\bot_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{\bot_2} \end{array}\right], \left[\begin{array}{c|c} \varepsilon_{\bot_1} & \mathbf{0} \\ \hline \mathbf{0} & \varepsilon_{\bot_2} \end{array}\right], \left[\begin{array}{c|c} \varepsilon_{\not\bot_1} & \mathbf{0} \\ \hline \mathbf{0} & \varepsilon_{\not\bot_2} \end{array}\right], \overrightarrow{F} \right)
$$

where

$$\overrightarrow{I} := \left[\begin{array}{c} \overrightarrow{I_1} \\ \hline \overrightarrow{I_2} \end{array}\right] \qquad \text{and} \qquad \overrightarrow{F} := \left[\begin{array}{c} \overrightarrow{F_1} \\ \hline \overrightarrow{F_2} \end{array}\right] \ .$$

First, note that

$$\vdash \mathbf{WM} = \left[\begin{array}{c|c} \mathbf{WM}_1 & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM}_2 \end{array}\right] \ . \tag{B.3}$$

The case $\geqslant$ follows from the fact that 0 is the minimum of the algebra and (3.26) and (3.30). For the case $\leqslant$, first recall that, by hypothesis, $S_1 \cap S_2 = \emptyset$ and $\Gamma_1 \cap \Gamma_2 = \emptyset$. By definition of $\delta$, any transition starting with $s_i \in S_1$ (respectively, $s_k \in S_2$) also ends with a state of $S_1$ (respectively, $S_2$). So, any block containing $s_i$ (respectively, $s_k$) as its starting label also has labels of $S_1$ (respectively, $S_2$) as other labels (that is to say, ending label, and call label and return label if it is a binary block). In other words,

- for every $s_i \in S_1$ and $\lfloor_{s_i}\, m \,\rceil^{s_j} \in (\mathcal{B}_1, \mathcal{B}_2)^1$, it is true that $\lfloor_{s_i}\, m \,\rceil^{s_j} \in \mathcal{B}_1^1$;
- for every $s_i \in S_1$ and $\lfloor_{s_i}\, c{\downarrow_z}{\uparrow^w}\, r \,\rceil^{s_j} \in (\mathcal{B}_1, \mathcal{B}_2)^2$, it is true that $\lfloor_{s_i}\, c{\downarrow_z}{\uparrow^w}\, r \,\rceil^{s_j} \in \mathcal{B}_1^2$;
- for every $s_k \in S_2$ and $\lfloor_{s_k}\, m \,\rceil^{s_l} \in (\mathcal{B}_1, \mathcal{B}_2)^1$, it is true that $\lfloor_{s_k}\, m \,\rceil^{s_l} \in \mathcal{B}_2^1$;
- for every $s_k \in S_2$ and $\lfloor_{s_k}\, c{\downarrow_z}{\uparrow^w}\, r \,\rceil^{s_l} \in (\mathcal{B}_1, \mathcal{B}_2)^2$, it is true that $\lfloor_{s_k}\, c{\downarrow_z}{\uparrow^w}\, r \,\rceil^{s_l} \in \mathcal{B}_2^2$.

Using the previous reasoning along with axiom (3.14), it is easy to prove all inequation of the case $\leqslant$ of (B.3). Let $s_{(s_i,s_j)} := \langle\!|_{s_i}\, \mathcal{B}_1 \,|\!\rangle^{s_j}$, $s_{(s_i,s_k)} := 0$, $s_{(s_k,s_i)} := 0$ and $s_{(s_k,s_l)} := \langle\!|_{s_k}\, \mathcal{B}_2 \,|\!\rangle^{s_l}$ for all $s_i, s_j \in S_1$ and $s_k, s_l \in S_2$. So, it suffices to prove, for all $s_i, s_j \in S_1$ and $s_k, s_l \in S_2$,

$$(\wedge\ m \mid \underset{s_i}{[}\ m\ \overset{s_j}{]} \in \mathcal{B}_1^1 : m \leqslant \underset{s_i}{\langle\!|}\ \mathcal{B}_1\ \overset{s_j}{|\!\rangle}) \ ,$$

$$(\wedge\ m, v \mid \underset{s_i}{[}\ m\ \overset{v}{]} \in \mathcal{B}_1^1 : m \cdot \underset{v}{\langle\!|}\ \mathcal{B}_1\ \overset{s_j}{|\!\rangle} \leqslant \underset{s_i}{\langle\!|}\ \mathcal{B}_1\ \overset{s_j}{|\!\rangle}) \ ,$$

$$(\wedge\ c, z, r, w \mid \underset{s_i}{[}\ c\underset{z}{\downarrow}\overset{w}{\uparrow} r\ \overset{s_j}{]} \in \mathcal{B}_1^2 : c \cdot \underset{z}{\langle\!|}\ \mathcal{B}_1\ \overset{w}{|\!\rangle} \cdot r \leqslant \underset{s_i}{\langle\!|}\ \mathcal{B}_1\ \overset{s_j}{|\!\rangle}) \ ,$$

$$(\wedge\ c, z, r, w, v \mid \underset{s_i}{[}\ c\underset{z}{\downarrow}\overset{w}{\uparrow} r\ \overset{v}{]} \in \mathcal{B}_1^2 : c \cdot \underset{z}{\langle\!|}\ \mathcal{B}_1\ \overset{w}{|\!\rangle} \cdot r \cdot \underset{v}{\langle\!|}\ \mathcal{B}_1\ \overset{s_j}{|\!\rangle} \leqslant \underset{s_i}{\langle\!|}\ \mathcal{B}_1\ \overset{s_j}{|\!\rangle}) \ ,$$

$$(\wedge\ m, v \mid \underset{s_i}{[}\ m\ \overset{v}{]} \in \mathcal{B}_1^1 : m \cdot 0 \leqslant 0) \ ,$$

$$(\wedge\ c, z, r, w, v \mid \underset{s_i}{[}\ c\underset{z}{\downarrow}\overset{w}{\uparrow} r\ \overset{v}{]} \in \mathcal{B}_1^2 : c \cdot \underset{z}{\langle\!|}\ \mathcal{B}_1\ \overset{w}{|\!\rangle} \cdot r \cdot 0 \leqslant 0) \ ,$$

$$(\wedge\ m, v \mid \underset{s_k}{[}\ m\ \overset{v}{]} \in \mathcal{B}_2^1 : m \cdot 0 \leqslant 0) \ ,$$

$$(\wedge\ c,z,r,w,v \mid [\underset{s_k}{\ } c\underset{z}{\downarrow}\overset{w}{\uparrow} r\overset{v}{\underset{s_k}{\ }}] \in \mathcal{B}_2^2 : c\cdot \underset{z}{(\!|}\mathcal{B}_2\overset{w}{|\!)}\cdot r\cdot 0 \leqslant 0)\ ,$$

$$(\wedge\ m \mid [\underset{s_k}{\ } m\overset{s_l}{\ }] \in \mathcal{B}_2^1 : m \leqslant \underset{s_k}{(\!|}\mathcal{B}_2\overset{s_l}{|\!)})\ ,$$

$$(\wedge\ m,v \mid [\underset{s_k}{\ } m\overset{v}{\ }] \in \mathcal{B}_2^1 : m\cdot \underset{v}{(\!|}\mathcal{B}_2\overset{s_l}{|\!)} \leqslant \underset{s_k}{(\!|}\mathcal{B}_2\overset{s_l}{|\!)})\ ,$$

$$(\wedge\ c,z,r,w \mid [\underset{s_k}{\ } c\underset{z}{\downarrow}\overset{w}{\uparrow} r\overset{s_l}{\ }] \in \mathcal{B}_2^2 : c\cdot \underset{z}{(\!|}\mathcal{B}_2\overset{w}{|\!)}\cdot r \leqslant \underset{s_k}{(\!|}\mathcal{B}_2\overset{s_l}{|\!)})\ ,$$

$$(\wedge\ c,z,r,w,v \mid [\underset{s_k}{\ } c\underset{z}{\downarrow}\overset{w}{\uparrow} r\overset{v}{\underset{s_k}{\ }}] \in \mathcal{B}_2^2 : c\cdot \underset{z}{(\!|}\mathcal{B}_2\overset{w}{|\!)}\cdot r\cdot \underset{v}{(\!|}\mathcal{B}_2\overset{s_l}{|\!)} \leqslant \underset{s_k}{(\!|}\mathcal{B}_2\overset{s_l}{|\!)})\ .$$

These inequations are easily proved by using zero of $\cdot$ and axioms (3.2), (3.3) and (3.4).

It remains to show (4.14).

$$\begin{aligned}
&\left[\ \overrightarrow{I_1}^t \ \middle|\ \overrightarrow{I_2}^t\ \right]\bullet\left(\left[\begin{array}{c|c}\mathbf{T}_{\perp_1} & \mathbf{0}\\ \hline \mathbf{0} & \mathbf{T}_{\perp_2}\end{array}\right] + \mathbf{WM} + \left[\begin{array}{c|c}\varepsilon_{\perp_1} & \mathbf{0}\\ \hline \mathbf{0} & \varepsilon_{\perp_2}\end{array}\right]\right.\\
&\qquad\left. + \left(\left[\begin{array}{c|c}\mathbf{T}_{\mathsf{c}_1} & \mathbf{0}\\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}_2}\end{array}\right]\bullet\mathbf{WM}\right)^+\bullet\left[\begin{array}{c|c}\varepsilon_{\not\perp_1} & \mathbf{0}\\ \hline \mathbf{0} & \varepsilon_{\not\perp_2}\end{array}\right]\right)^\star\\
&\quad\bullet\left(\left[\begin{array}{c|c}\mathbf{T}_{\mathsf{c}_1} & \mathbf{0}\\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}_2}\end{array}\right]\bullet\mathbf{WM}\right)^\star\bullet\left[\begin{array}{c}\overrightarrow{F_1}\\ \hline \overrightarrow{F_2}\end{array}\right]\\
=&\quad \{\!\!\{\ \text{Equation (B.3)}\ \}\!\!\}\\
&\left[\ \overrightarrow{I_1}^t \ \middle|\ \overrightarrow{I_2}^t\ \right]\bullet\left(\left[\begin{array}{c|c}\mathbf{T}_{\perp_1} & \mathbf{0}\\ \hline \mathbf{0} & \mathbf{T}_{\perp_2}\end{array}\right] + \left[\begin{array}{c|c}\mathbf{WM}_1 & \mathbf{0}\\ \hline \mathbf{0} & \mathbf{WM}_2\end{array}\right] + \left[\begin{array}{c|c}\varepsilon_{\perp_1} & \mathbf{0}\\ \hline \mathbf{0} & \varepsilon_{\perp_2}\end{array}\right]\right.\\
&\qquad\left. + \left(\left[\begin{array}{c|c}\mathbf{T}_{\mathsf{c}_1} & \mathbf{0}\\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}_2}\end{array}\right]\bullet\left[\begin{array}{c|c}\mathbf{WM}_1 & \mathbf{0}\\ \hline \mathbf{0} & \mathbf{WM}_2\end{array}\right]\right)^+\bullet\left[\begin{array}{c|c}\varepsilon_{\not\perp_1} & \mathbf{0}\\ \hline \mathbf{0} & \varepsilon_{\not\perp_2}\end{array}\right]\right)^\star\\
&\quad\bullet\left(\left[\begin{array}{c|c}\mathbf{T}_{\mathsf{c}_1} & \mathbf{0}\\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}_2}\end{array}\right]\bullet\left[\begin{array}{c|c}\mathbf{WM}_1 & \mathbf{0}\\ \hline \mathbf{0} & \mathbf{WM}_2\end{array}\right]\right)^\star\bullet\left[\begin{array}{c}\overrightarrow{F_1}\\ \hline \overrightarrow{F_2}\end{array}\right]\\
=&\quad \{\!\!\{\ \text{Definition of } \bullet, {}^\star \text{ and } {}^+ \text{ \& Kleene algebra}\ \}\!\!\}\\
&\left[\ \overrightarrow{I_1}^t \ \middle|\ \overrightarrow{I_2}^t\ \right]\bullet\left(\left[\begin{array}{c|c}\mathbf{T}_{\perp_1} & \mathbf{0}\\ \hline \mathbf{0} & \mathbf{T}_{\perp_2}\end{array}\right] + \left[\begin{array}{c|c}\mathbf{WM}_1 & \mathbf{0}\\ \hline \mathbf{0} & \mathbf{WM}_2\end{array}\right] + \left[\begin{array}{c|c}\varepsilon_{\perp_1} & \mathbf{0}\\ \hline \mathbf{0} & \varepsilon_{\perp_2}\end{array}\right]\right.\\
&\qquad\left. + \left[\begin{array}{c|c}(\mathbf{T}_{\mathsf{c}_1}\bullet\mathbf{WM}_1)^+\bullet\varepsilon_{\not\perp_1} & \mathbf{0}\\ \hline \mathbf{0} & (\mathbf{T}_{\mathsf{c}_2}\bullet\mathbf{WM}_2)^+\bullet\varepsilon_{\not\perp_2}\end{array}\right]\right)^\star\\
&\quad\bullet\left[\begin{array}{c|c}(\mathbf{T}_{\mathsf{c}_1}\bullet\mathbf{WM}_1)^\star & \mathbf{0}\\ \hline \mathbf{0} & (\mathbf{T}_{\mathsf{c}_2}\bullet\mathbf{WM}_2)^\star\end{array}\right]\bullet\left[\begin{array}{c}\overrightarrow{F_1}\\ \hline \overrightarrow{F_2}\end{array}\right]\\
=&\quad \{\!\!\{\ \text{Definition of } + \text{ and } {}^\star \text{ \& Kleene algebra}\ \}\!\!\}
\end{aligned}$$

$$
\begin{aligned}
&\left[\begin{array}{c|c} \overrightarrow{I_1}^t & \overrightarrow{I_2}^t \end{array}\right] \bullet \left[\begin{array}{c|c} (\mathbf{T}_{\perp_1} + \mathbf{WM}_1 + \varepsilon_{\perp_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\perp_1})^\star & \mathbf{0} \\ \hline \mathbf{0} & (\mathbf{T}_{\perp_2} + \mathbf{WM}_2 + \varepsilon_{\perp_2} + (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^+ \bullet \varepsilon_{\not\perp_2})^\star \end{array}\right] \\
&\quad \bullet \left[\begin{array}{c|c} (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star & \mathbf{0} \\ \hline \mathbf{0} & (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^\star \end{array}\right] \bullet \left[\begin{array}{c} \overrightarrow{F_1} \\ \hline \overrightarrow{F_2} \end{array}\right] \\
= \quad & \{\!\{ \text{ Definition of } \bullet \text{ \& Zero of } \cdot \text{ \& Identity of } + \text{ } \}\!\} \\
&\overrightarrow{I_1}^t \bullet (\mathbf{T}_{\perp_1} + \mathbf{WM}_1 + \varepsilon_{\perp_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\perp_1})^\star \bullet (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \\
&+ \overrightarrow{I_2}^t \bullet (\mathbf{T}_{\perp_2} + \mathbf{WM}_2 + \varepsilon_{\perp_2} + (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^+ \bullet \varepsilon_{\not\perp_2})^\star \bullet (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^\star \bullet \overrightarrow{F_2} \\
= \quad & \{\!\{ \text{ Induction hypotheses } \}\!\} \\
&q_1 + q_2
\end{aligned}
$$

## B.7 Induction Case: Operator $\cdot$

Let $p := q_1 \cdot q_2$ where $q_1$ and $q_2$ are visibly pushdown regular expressions. Suppose there exists a semi-visibly pushdown automaton $\mathcal{A}_1 := (S_1, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_1 \cup \{\perp\}, \delta_1, I_1, F_1)$ that accepts the language $\mathcal{L}(q_1)$ and the algebraic encoding of $\mathcal{A}_1$ by the structure $(\overrightarrow{I_1}, \mathbf{WM}_1, \mathbf{T}_{\mathsf{c}_1}, \mathbf{T}_{\perp_1}, \varepsilon_{\perp_1}, \varepsilon_{\not\perp_1}, \overrightarrow{F_1})$ is such that (4.14) is valid. Also, suppose there exists a semi-visibly pushdown automaton $\mathcal{A}_2 := (S_2, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_2 \cup \{\perp\}, \delta_2, I_2, F_2)$ that accepts the language $\mathcal{L}(q_2)$ and the algebraic encoding of $\mathcal{A}_2$ by the structure $(\overrightarrow{I_2}, \mathbf{WM}_2, \mathbf{T}_{\mathsf{c}_2}, \mathbf{T}_{\perp_2}, \varepsilon_{\perp_2}, \varepsilon_{\not\perp_2}, \overrightarrow{F_2})$ is such that (4.14) is valid.

Without loss of generality, suppose $S_1 \cap S_2 = \emptyset$ and $\Gamma_1 \cap \Gamma_2 = \emptyset$. Define the semi-visibly pushdown automaton

$$\mathcal{A} := (S_1 \cup S_2, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, (\Gamma_1 \cup \Gamma_2) \cup \{\perp\}, \delta, I_1, F_2)$$

where $\delta := \delta_1 \cup \delta_2 \cup \{(f, \varepsilon, d; i', \perp) \mid f \in F_1 \wedge i' \in I_2 \wedge d \in (\Gamma_1 \cup \Gamma_2) \cup \{\perp\}\} \cup \mathsf{Waste}$, and

$$
\begin{aligned}
\mathsf{Waste} := \quad & \{(s, \varepsilon, d; s', \perp) \mid d \in \Gamma_2 \wedge (\exists\ d' \mid d' \in \Gamma_1 : (s, \varepsilon, d'; s', \perp) \in \delta_1)\} \\
& \cup \{(s, \varepsilon, d; s', \perp) \mid d \in \Gamma_1 \wedge (\exists\ d' \mid d' \in \Gamma_2 : (s, \varepsilon, d'; s', \perp) \in \delta_2)\} \ .
\end{aligned}
$$

It is easy to see that $\mathcal{A}$ accepts the language $\mathcal{L}(q_1 \cdot q_2)$. Note that the elements of $\mathsf{Waste}$ are added just to satisfy the property (2.3).

Let $\mathcal{B} := \mathcal{B}_1, \mathcal{B}_2$ (this list of block is used since $S_1 \cap S_2 = \emptyset$ and $\Gamma_1 \cap \Gamma_2 = \emptyset$ by hypothesis, and the construction just adds $\varepsilon$-transitions). The algebraic encoding of

this automaton is

$$\left(\overrightarrow{I}, \mathbf{WM}, \left[\begin{array}{c|c} \mathbf{T}_{c_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{c_2} \end{array}\right], \left[\begin{array}{c|c} \mathbf{T}_{\perp_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{\perp_2} \end{array}\right], \left[\begin{array}{c|c} \varepsilon_{\perp_1} & \overrightarrow{F_1} \bullet \overrightarrow{I_2}^t \\ \hline \mathbf{0} & \varepsilon_{\perp_2} \end{array}\right], \left[\begin{array}{c|c} \varepsilon_{\not\perp_1} & \overrightarrow{F_1} \bullet \overrightarrow{I_2}^t \\ \hline \mathbf{0} & \varepsilon_{\not\perp_2} \end{array}\right], \overrightarrow{F}\right)$$

where

$$\overrightarrow{I} := \left[\begin{array}{c} \overrightarrow{I_1} \\ \hline \overrightarrow{0} \end{array}\right] \qquad \text{and} \qquad \overrightarrow{F} := \left[\begin{array}{c} \overrightarrow{0} \\ \hline \overrightarrow{F_2} \end{array}\right] \quad .$$

First, note that

$$\vdash \mathbf{WM} = \left[\begin{array}{c|c} \mathbf{WM}_1 & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM}_2 \end{array}\right] \quad . \tag{B.4}$$

The proof of (B.4) is similar to the proof of (B.3). The only difference is that we use the fact that, by definition of $\delta$, every *non-$\varepsilon$-transition* starting with $s_i \in S_1$ (respectively, $s_k \in S_2$) also ends with a state of $S_1$ (respectively, $S_2$). So, every block containing $s_i$ (respectively, $s_k$) as its starting label also has labels of $S_1$ (respectively, $S_2$) as other labels (that is to say ending label, and call label and return label if it is a binary block).

It remains to show (4.14).

$$\begin{aligned}
&\left[\begin{array}{c|c} \overrightarrow{I_1}^t & \overrightarrow{0} \end{array}\right] \bullet \left( \left[\begin{array}{c|c} \mathbf{T}_{\perp_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{\perp_2} \end{array}\right] + \mathbf{WM} + \left[\begin{array}{c|c} \varepsilon_{\perp_1} & \overrightarrow{F_1} \bullet \overrightarrow{I_2}^t \\ \hline \mathbf{0} & \varepsilon_{\perp_2} \end{array}\right] \right. \\
&\qquad \left. + \left( \left[\begin{array}{c|c} \mathbf{T}_{c_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{c_2} \end{array}\right] \bullet \mathbf{WM} \right)^+ \bullet \left[\begin{array}{c|c} \varepsilon_{\not\perp_1} & \overrightarrow{F_1} \bullet \overrightarrow{I_2}^t \\ \hline \mathbf{0} & \varepsilon_{\not\perp_2} \end{array}\right] \right)^\star \\
&\quad \bullet \left( \left[\begin{array}{c|c} \mathbf{T}_{c_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{c_2} \end{array}\right] \bullet \mathbf{WM} \right)^\star \bullet \left[\begin{array}{c} \overrightarrow{0} \\ \hline \overrightarrow{F_2} \end{array}\right] \\
= \quad & \{\!\{ \text{ Equation (B.4) } \}\!\} \\
&\left[\begin{array}{c|c} \overrightarrow{I_1}^t & \overrightarrow{0} \end{array}\right] \bullet \left( \left[\begin{array}{c|c} \mathbf{T}_{\perp_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{\perp_2} \end{array}\right] + \left[\begin{array}{c|c} \mathbf{WM}_1 & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM}_2 \end{array}\right] + \left[\begin{array}{c|c} \varepsilon_{\perp_1} & \overrightarrow{F_1} \bullet \overrightarrow{I_2}^t \\ \hline \mathbf{0} & \varepsilon_{\perp_2} \end{array}\right] \right. \\
&\qquad \left. + \left( \left[\begin{array}{c|c} \mathbf{T}_{c_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{c_2} \end{array}\right] \bullet \left[\begin{array}{c|c} \mathbf{WM}_1 & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM}_2 \end{array}\right] \right)^+ \bullet \left[\begin{array}{c|c} \varepsilon_{\not\perp_1} & \overrightarrow{F_1} \bullet \overrightarrow{I_2}^t \\ \hline \mathbf{0} & \varepsilon_{\not\perp_2} \end{array}\right] \right)^\star \\
&\quad \bullet \left( \left[\begin{array}{c|c} \mathbf{T}_{c_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{c_2} \end{array}\right] \bullet \left[\begin{array}{c|c} \mathbf{WM}_1 & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM}_2 \end{array}\right] \right)^\star \bullet \left[\begin{array}{c} \overrightarrow{0} \\ \hline \overrightarrow{F_2} \end{array}\right] \\
= \quad & \{\!\{ \text{ Definition of } \bullet, {}^+ \text{ and } {}^\star \text{ \& Kleene algebra } \}\!\}
\end{aligned}$$

$$
\begin{aligned}
&\left[\begin{array}{c|c} \overrightarrow{I_1}^t & \overrightarrow{0} \end{array}\right] \bullet \left( \left[\begin{array}{c|c} \mathbf{T}_{\perp_1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{T}_{\perp_2} \end{array}\right] + \left[\begin{array}{c|c} \mathbf{WM}_1 & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM}_2 \end{array}\right] + \left[\begin{array}{c|c} \varepsilon_{\perp_1} & \overrightarrow{F_1} \bullet \overrightarrow{I_2}^t \\ \hline \mathbf{0} & \varepsilon_{\perp_2} \end{array}\right] \right. \\
&\qquad \left. + \left[\begin{array}{c|c} (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\perp_1} & (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \overrightarrow{F_1} \bullet \overrightarrow{I_2}^t \\ \hline \mathbf{0} & (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^+ \bullet \varepsilon_{\not\perp_2} \end{array}\right] \right)^\star \\
&\quad \bullet \left[\begin{array}{c|c} (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star & \mathbf{0} \\ \hline \mathbf{0} & (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^\star \end{array}\right] \bullet \left[\begin{array}{c} \overrightarrow{0} \\ \hline \overrightarrow{F_2} \end{array}\right] \\
= \quad & \{\!\{ \text{ Definition of } + \text{ and } {}^\star \text{ \& Kleene algebra (including: } 1 + q^+ = q^*) \text{ } \}\!\} \\
&\left[\begin{array}{c|c} \overrightarrow{I_1}^t & \overrightarrow{0} \end{array}\right] \bullet \left[\begin{array}{c|c} (\mathbf{T}_{\perp_1} + \mathbf{WM}_1 + \varepsilon_{\perp_1} \\ \quad + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\perp_1})^\star & \begin{array}{l} (\mathbf{T}_{\perp_1} + \mathbf{WM}_1 + \varepsilon_{\perp_1} \\ \quad + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\perp_1})^\star \\ \bullet (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \bullet \overrightarrow{I_2}^t \\ \bullet (\mathbf{T}_{\perp_2} + \mathbf{WM}_2 + \varepsilon_{\perp_2} \\ \quad + (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^+ \bullet \varepsilon_{\not\perp_2})^\star \end{array} \\ \hline \mathbf{0} & \begin{array}{l} (\mathbf{T}_{\perp_2} + \mathbf{WM}_2 + \varepsilon_{\perp_2} \\ \quad + (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^+ \bullet \varepsilon_{\not\perp_2})^\star \end{array} \end{array}\right] \\
&\quad \bullet \left[\begin{array}{c|c} (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star & \mathbf{0} \\ \hline \mathbf{0} & (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^\star \end{array}\right] \bullet \left[\begin{array}{c} \overrightarrow{0} \\ \hline \overrightarrow{F_2} \end{array}\right] \\
= \quad & \{\!\{ \text{ Definition of } \bullet \text{ \& Zero of } \cdot \text{ \& Identity of } + \text{ } \}\!\} \\
&\overrightarrow{I_1}^t \bullet (\mathbf{T}_{\perp_1} + \mathbf{WM}_1 + \varepsilon_{\perp_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\perp_1})^\star \bullet (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \\
&\cdot \overrightarrow{I_2}^t \bullet (\mathbf{T}_{\perp_2} + \mathbf{WM}_2 + \varepsilon_{\perp_2} + (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^+ \bullet \varepsilon_{\not\perp_2})^\star \bullet (\mathbf{T}_{\mathsf{c}_2} \bullet \mathbf{WM}_2)^\star \bullet \overrightarrow{F_2} \\
= \quad & \{\!\{ \text{ Induction hypotheses } \}\!\} \\
&q_1 \cdot q_2
\end{aligned}
$$

## B.8 Induction Case: Operator $^*$

Let $p := q_1^*$ where $q_1$ is a visibly pushdown regular expression. Suppose there exists a semi-visibly pushdown automaton $\mathcal{A}_1 := (S_1, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_1 \cup \{\perp\}, \delta_1, I_1, F_1)$ that accepts the language $\mathcal{L}(q_1)$ and the algebraic encoding of $\mathcal{A}_1$ by the structure $(\overrightarrow{I_1}, \mathbf{WM}_1, \mathbf{T}_{\mathsf{c}_1}, \mathbf{T}_{\perp_1}, \varepsilon_{\perp_1}, \varepsilon_{\not\perp_1}, \overrightarrow{F_1})$ is such that (4.14) is valid.

Since $q_1^* = 1 + q_1 \cdot q_1^*$ by Kleene algebra and since the cases 1 and $+$ are already proved, it suffices to find a semi-visibly pushdown automaton that accepts $\mathcal{L}(q_1 \cdot q_1^*)$ and such that (4.14) is valid. Define the semi-visibly pushdown automaton

$$\mathcal{A} := (S_1, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_1 \cup \{\perp\}, \delta, I_1, F_1)$$

where

$$\delta := \delta_1 \cup \{(f, \varepsilon, d; i, \perp) \mid f \in F_1 \wedge i \in I_1 \wedge d \in \Gamma_1 \cup \{\perp\}\} \ .$$

It is easy to see that $\mathcal{A}$ accepts the language $\mathcal{L}(q_1 \cdot q_1^*)$.

Let $\mathcal{B} := \mathcal{B}_1$ (this list of block is used since the construction just adds $\varepsilon$-transitions between final states and initial states). The algebraic encoding of this automaton is

$$(\overrightarrow{I_1}, \mathbf{WM}_1, \mathbf{T}_{\mathsf{c}_1}, \mathbf{T}_{\perp_1}, \varepsilon_{\perp_1} + \overrightarrow{F_1} \bullet \overrightarrow{I_1}^t, \varepsilon_{\not\perp_1} + \overrightarrow{F_1} \bullet \overrightarrow{I_1}^t, \overrightarrow{F_1}) \ .$$

It remains to show (4.14).

$$\begin{aligned}
& \overrightarrow{I_1}^t \bullet (\mathbf{T}_{\perp_1} + \mathbf{WM}_1 + \varepsilon_{\perp_1} + \overrightarrow{F_1} \bullet \overrightarrow{I_1}^t + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet (\varepsilon_{\not\perp_1} + \overrightarrow{F_1} \bullet \overrightarrow{I_1}^t))^\star \bullet (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \\
= & \qquad \{\!\{ \text{ Distributivity of } \cdot \text{ over } + \text{ \& Commutativity of } + \ \}\!\} \\
& \overrightarrow{I_1}^t \bullet (\overrightarrow{F_1} \bullet \overrightarrow{I_1}^t + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \overrightarrow{F_1} \bullet \overrightarrow{I_1}^t + \mathbf{T}_{\perp_1} + \mathbf{WM}_1 + \varepsilon_{\perp_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\perp_1})^\star \bullet (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \\
= & \qquad \{\!\{ \text{ Kleene algebra (mainly, } 1 + q^+ = q^*) \ \}\!\} \\
& \overrightarrow{I_1}^t \bullet ((\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \bullet \overrightarrow{I_1}^t + \mathbf{T}_{\perp_1} + \mathbf{WM}_1 + \varepsilon_{\perp_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\perp_1})^\star \bullet (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \\
= & \qquad \{\!\{ \text{ Kleene algebra: Denesting rule } \}\!\} \\
& \overrightarrow{I_1}^t \bullet (\mathbf{T}_{\perp_1} + \mathbf{WM}_1 + \varepsilon_{\perp_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\perp_1})^\star \\
& \bullet \left( (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \bullet \overrightarrow{I_1}^t \bullet (\mathbf{T}_{\perp_1} + \mathbf{WM}_1 + \varepsilon_{\perp_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\perp_1})^\star \right)^\star \bullet \\
& (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \\
= & \qquad \{\!\{ \text{ Kleene algebra: Sliding rule } \}\!\} \\
& \overrightarrow{I_1}^t \bullet (\mathbf{T}_{\perp_1} + \mathbf{WM}_1 + \varepsilon_{\perp_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\perp_1})^\star \bullet (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \\
& \cdot \left( \overrightarrow{I_1}^t \bullet (\mathbf{T}_{\perp_1} + \mathbf{WM}_1 + \varepsilon_{\perp_1} + (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^+ \bullet \varepsilon_{\not\perp_1})^\star \bullet (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star \bullet \overrightarrow{F_1} \right)^\star \\
= & \qquad \{\!\{ \text{ Induction hypothesis } \}\!\} \\
& q_1 \cdot q_1^*
\end{aligned}$$

## B.9 Induction Case: Family of Operators $\mathcal{G}$

Let $V$ be a finite set of labels and $x, y \in V$ be labels. Let $\mathcal{B}$ be a finite list of unary blocks each containing one element of $\Sigma_{\mathbf{i}} \cup \{0, 1\}$, and binary blocks each containing one element of $\Sigma_{\mathbf{c}}$ as left operand and one element of $\Sigma_{\mathbf{r}}$ as right operand, where all blocks use labels from $V$. So, the expression that must be encoded in a semi-visibly pushdown automaton is $(\!|_x \mathcal{B} |\!)^y$.

We first suppose (we will prove it after) that there exists a finite set of expressions $\{(\!|_{x'}\, \mathcal{C}\, |\!)^y\}_{x' \in I}$ where $I \subseteq V \cup \{z_{\mathfrak{c}} \mid z \in V\}$ and the list $\mathcal{C}$ has the following properties:

i. the set of labels used by $\mathcal{C}$ is $V'$ and is at most $V \cup \{z_{\mathfrak{c}} \mid z \in V\}$;

ii. there is no unary block of the form $[_z\, 0\, ]^{z'}$ in $\mathcal{C}^1$, where $z, z' \in V'$;

iii. there is exactly one unary block of the form $[_z\, 1\, ]^z$ in $\mathcal{C}^1$ for each $z \in V'$;

iv. there is no other block of the form $[_z\, 1\, ]^{z'}$ in $\mathcal{C}^1$ except those defined by iii;

such that

$$\vdash \underset{x}{(\!|}\, \mathcal{B}\, \underset{}{|\!)^y} = (\sum x' \mid x' \in I : \underset{x'}{(\!|}\, \mathcal{C}\, \underset{}{|\!)^y}) \ .$$

This assumption is in fact valid for every expression $(\!|_x\, \mathcal{B}\, |\!)^y$ as we will show below in Lemma B.1. Moreover, the size of $(\sum x' \mid x' \in I : (\!|_{x'}\, \mathcal{C}\, |\!)^y)$ will be polynomial in the size of $(\!|_x\, \mathcal{B}\, |\!)^y$. However, since the proof of Lemma B.1 is very involved, we first finish the proof of the inductive case for the family of operators $\mathcal{G}$ before going on to the definition and proof of Lemma B.1.

Define the semi-visibly pushdown automaton

$$\mathcal{A} := (V' \cup \{f'\}, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, V' \times \Sigma_{\mathbf{r}} \times V' \cup \{\bot\}, \delta, I, \{f'\})$$

where $f' \notin V'$ and

$$\begin{aligned} \delta := \ & \{(w, a, \lambda; w', \lambda) \mid [_w\, a\, ]^{w'} \in \mathcal{C}^1\} \\ & \cup \{(w, c, \lambda; w', (z, r, z')), (z, r, (z, r, z'); z', \lambda) \mid [_w\, c \downarrow_{w'} \uparrow^z r\, ]^{z'} \in \mathcal{C}^2\} \\ & \cup \{(y, \varepsilon, \bot; f', \bot)\} \ . \end{aligned}$$

It is easy to see that $\mathcal{A}$ accepts the language $\mathcal{L}((\sum x' \mid x' \in I : (\!|_{x'}\, \mathcal{C}\, |\!)^y))$.

Let $\overrightarrow{F}$ be the column vector of size $|V'|$ such that 1 is in row $y$ and 0 in every other row. Let $\mathcal{C}' := \mathcal{C}, [_{f'}\, 1\, ]^{f'}$. The algebraic encoding of this automaton is

$$\left( \left[ \begin{array}{c} \overrightarrow{I} \\ \hline 0 \end{array} \right], \mathbf{WM}, \mathbf{T}_{\mathbf{c}}, \mathbf{0}, \left[ \begin{array}{c|c} \mathbf{0} & \overrightarrow{F} \\ \hline 0 & 0 \end{array} \right], \mathbf{0}, \left[ \begin{array}{c} \overrightarrow{0} \\ \hline 1 \end{array} \right] \right) \ .$$

It is easy to see that

$$\vdash \mathbf{WM} = \left[ \begin{array}{c|c} \mathbf{WM}_1 & \overrightarrow{0} \\ \hline 0 & 1 \end{array} \right] \tag{B.5}$$

where $\mathbf{WM}_1$ is a matrix of size $|V'| \times |V'|$ containing, for each entry $(j_1, j_2)$, the expression $(\!|_{j_1} \mathcal{C} |\!)^{j_2}$. The proof of (B.5) uses (B.1) and (B.3) (they are applicable in this case),

$$\left[\begin{array}{c|c} \mathbf{WM}_1 & \overrightarrow{0} \\ \hline 0 & 1 \end{array}\right] = \left[\begin{array}{c|c} \mathbf{WM}_1 & \overrightarrow{0} \\ \hline 0 & (\!|_{f'} [_{f'}\, 1 ]^{f'} |\!)^{f'} \end{array}\right] = \mathbf{WM} \ .$$

Also, by definition of $\mathbf{T}_\mathsf{c}$, it is direct that every entry containing $f'$ is 0. In other words,

$$\vdash \mathbf{T}_\mathsf{c} = \left[\begin{array}{c|c} \mathbf{T}_{\mathsf{c}_1} & \overrightarrow{0} \\ \hline 0 & 0 \end{array}\right] \tag{B.6}$$

where $\mathbf{T}_{\mathsf{c}_1}$ is a matrix of size $|V'| \times |V'|$ containing, for each entry $(j_1, j_2)$, the expression $\mathbf{T}_\mathsf{c}[j_1, j_2]$.

It remains to show (4.14).

$$\begin{array}{ll}
 & \left[\begin{array}{c|c} \overrightarrow{I}^t & 0 \end{array}\right] \bullet \left( \mathbf{0} + \mathbf{WM} + \left[\begin{array}{c|c} \mathbf{0} & \overrightarrow{F} \\ \hline 0 & 0 \end{array}\right] + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \mathbf{0} \right)^\star \bullet (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^\star \bullet \left[\begin{array}{c} \overrightarrow{0} \\ \hline 1 \end{array}\right] \\
= & \quad \{\!\{ \text{Identity of } \cdot \text{ \& Zero of } + \text{ \& Equations (B.5) and (B.6)} \}\!\} \\
 & \left[\begin{array}{c|c} \overrightarrow{I}^t & 0 \end{array}\right] \bullet \left( \left[\begin{array}{c|c} \mathbf{WM}_1 & \overrightarrow{0} \\ \hline 0 & 1 \end{array}\right] + \left[\begin{array}{c|c} \mathbf{0} & \overrightarrow{F} \\ \hline 0 & 0 \end{array}\right] \right)^\star \bullet \left( \left[\begin{array}{c|c} \mathbf{T}_{\mathsf{c}_1} & \overrightarrow{0} \\ \hline 0 & 0 \end{array}\right] \bullet \left[\begin{array}{c|c} \mathbf{WM}_1 & \overrightarrow{0} \\ \hline 0 & 1 \end{array}\right] \right)^\star \bullet \\
 & \left[\begin{array}{c} \overrightarrow{0} \\ \hline 1 \end{array}\right] \\
= & \quad \{\!\{ \text{Definition of } +, \bullet \text{ and } \star \text{ \& Kleene algebra} \}\!\} \\
 & \left[\begin{array}{c|c} \overrightarrow{I}^t & 0 \end{array}\right] \bullet \left[\begin{array}{c|c} \mathbf{WM}_1^\star & \mathbf{WM}_1^\star \bullet \overrightarrow{F} \\ \hline 0 & 1 \end{array}\right] \bullet \left[\begin{array}{c|c} (\mathbf{T}_{\mathsf{c}_1} \bullet \mathbf{WM}_1)^\star & \overrightarrow{0} \\ \hline 0 & 1 \end{array}\right] \bullet \left[\begin{array}{c} \overrightarrow{0} \\ \hline 1 \end{array}\right] \\
= & \quad \{\!\{ \text{Definition of } \bullet \text{ \& Kleene algebra} \}\!\} \\
 & \overrightarrow{I}^t \bullet \mathbf{WM}_1^\star \bullet \overrightarrow{F} \\
= & \quad \{\!\{ \text{Lemma 4.5, property iii \& Definition of } \overrightarrow{I}, \overrightarrow{F} \text{ and } \mathbf{WM}_1 \text{ \& Definition of } \bullet \}\!\} \\
 & (\sum x' \mid x' \in I : (\!|_{x'} \mathcal{C} |\!)^y)
\end{array}$$

### B.9.1 Simplification of $(\!|\,|\!)$-Expressions

Now, it remains to state and prove Lemma B.1.

**Lemma B.1** (Simplification of $(\!|\,|\!)$-expressions)**.** *Let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be three disjoint finite sets such that at least one of them is nonempty. Let $V$ be a finite set of labels and $x, y \in V$ be labels. Let $\mathcal{B}$ be a finite list of unary blocks each containing one element of $\Sigma_{\mathbf{i}} \cup \{0,1\}$, and binary blocks each containing one element of $\Sigma_{\mathbf{c}}$ as left operand and one element of $\Sigma_{\mathbf{r}}$ as right operand, where all blocks use labels from $V$. Then, there exists a finite set of expressions $\{(\!|_{x'}\, \mathcal{C}\,|\!)^y\}_{x' \in I}$ where $I \subseteq V \cup \{z_{\mathfrak{c}} \mid z \in V\}$ and the list $\mathcal{C}$ has the following properties:*

*i. the set of labels used by $\mathcal{C}$ is $V'$ and is at most $V \cup \{z_{\mathfrak{c}} \mid z \in V\}$;*

*ii. there is no unary block of the form $[_z\, 0\,]^{z'}$ in $\mathcal{C}^1$, where $z, z' \in V'$;*

*iii. there is exactly one unary block of the form $[_z\, 1\,]^z$ in $\mathcal{C}^1$ for each $z \in V'$;*

*iv. there is no other block of the form $[_z\, 1\,]^{z'}$ in $\mathcal{C}^1$ except those defined by iii;*

*such that*

$$\vdash (\!|\, \mathcal{B}\,|\!)^y_x = (\sum\, x' \mid x' \in I : (\!|\, \mathcal{C}\,|\!)^y_{x'}) \ .$$

*Moreover, the size of $(\sum\, x' \mid x' \in I : (\!|_{x'}\, \mathcal{C}\,|\!)^y)$ is polynomial in the size of $(\!|_x\, \mathcal{B}\,|\!)^y$.*

*Proof.* First note that the set $V$ can be restricted to the "effectively used labels" in $V$. In fact, there is at most $2 \times |\mathcal{B}^1| + 4 \times |\mathcal{B}^2| + 2$ labels in the set of effectively used labels in $V$. Without loss of generality, suppose that $V$ is exactly the set of effectively used labels in $V$. So, the size of $V$ is polynomial in the number of the operands used in $(\!|_x\, \mathcal{B}\,|\!)^y$.

The proof is done by four successive transformations of the expression $(\!|_x\, \mathcal{B}\,|\!)^y$.

**Transformation 1: Clone the Labels from $V$**

We first "clone" all labels from $V$ by defining the set of labels $\{z_{\mathfrak{c}} \mid z \in V\}$ in which each $z_{\mathfrak{c}}$ is a fresh label. Let $V' := V \cup \{z_{\mathfrak{c}} \mid z \in V\}$. Define the finite list $\mathcal{B}'$ by the following rules:

a. for $a \in \Sigma_{\mathbf{i}} \cup \{0,1\}$ and $z, z' \in V$, the unary blocks $[_z\, a\,]^{z'}$ and $[_{z_{\mathfrak{c}}}\, a\,]^{z'}$ are in $(\mathcal{B}')^1$ if $[_z\, a\,]^{z'}$ is in $\mathcal{B}^1$;

b. for $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $z, z', w, w' \in V$, the binary blocks $[_z\, c{\downarrow_{w_{\mathfrak{c}}}}{\uparrow^{w'}}\, r\,]^{z'}$ and $[_{z_{\mathfrak{c}}}\, c{\downarrow_{w_{\mathfrak{c}}}}{\uparrow^{w'}}\, r\,]^{z'}$ are in $(\mathcal{B}')^2$ if $[_z\, c{\downarrow_{w}}{\uparrow^{w'}}\, r\,]^{z'}$ is in $\mathcal{B}^2$.

It is easy to see that for every $z, z' \in V$

$$\vdash \underset{z}{(\!|}\, \mathcal{B}' \,\overset{z'}{|\!)} = \underset{z_{\mathfrak{c}}}{(\!|}\, \mathcal{B}' \,\overset{z'}{|\!)} \quad . \tag{B.7}$$

We prove it.

$$\begin{aligned}
& (\!|_{z_{\mathfrak{c}}}\, \mathcal{B}' \,|\!)^{z'} \\
= & \qquad \{\!\{ \text{ Equation (3.22) } \}\!\} \\
& (\sum m \mid [_{z_{\mathfrak{c}}}\, m\,]^{z'} \in (\mathcal{B}')^1 : m) \\
& + (\sum m, v \mid [_{z_{\mathfrak{c}}}\, m\,]^{v} \in (\mathcal{B}')^1 : m \cdot (\!|_v\, \mathcal{B}' \,|\!)^{z'}) \\
& + (\sum c, w, r, w' \mid [_{z_{\mathfrak{c}}}\, c{\downarrow_{w_{\mathfrak{c}}}}{\uparrow^{w'}}\, r\,]^{z'} \in (\mathcal{B}')^2 : c \cdot (\!|_{w_{\mathfrak{c}}}\, \mathcal{B}' \,|\!)^{w'} \cdot r) \\
& + (\sum c, w, r, w', v \mid [_{z_{\mathfrak{c}}}\, c{\downarrow_{w_{\mathfrak{c}}}}{\uparrow^{w'}}\, r\,]^{v} \in (\mathcal{B}')^2 : c \cdot (\!|_{w_{\mathfrak{c}}}\, \mathcal{B}' \,|\!)^{w'} \cdot r \cdot (\!|_v\, \mathcal{B}' \,|\!)^{z'}) \\
= & \qquad \{\!\{ \text{ Definition of } \mathcal{B}', \text{ conditions a and b: } [_{z_{\mathfrak{c}}}\, m\,]^{v} \in (\mathcal{B}')^1 \Leftrightarrow [_z\, m\,]^{v} \in (\mathcal{B}')^1 \text{ and} \\
& \qquad\quad [_{z_{\mathfrak{c}}}\, c{\downarrow_{w_{\mathfrak{c}}}}{\uparrow^{w'}}\, r\,]^{v} \in (\mathcal{B}')^2 \Leftrightarrow [_z\, c{\downarrow_{w_{\mathfrak{c}}}}{\uparrow^{w'}}\, r\,]^{v} \in (\mathcal{B}')^2 \ \}\!\} \\
& (\sum m \mid [_z\, m\,]^{z'} \in (\mathcal{B}')^1 : m) \\
& + (\sum m, v \mid [_z\, m\,]^{v} \in (\mathcal{B}')^1 : m \cdot (\!|_v\, \mathcal{B}' \,|\!)^{z'}) \\
& + (\sum c, w, r, w' \mid [_z\, c{\downarrow_{w_{\mathfrak{c}}}}{\uparrow^{w'}}\, r\,]^{z'} \in (\mathcal{B}')^2 : c \cdot (\!|_{w_{\mathfrak{c}}}\, \mathcal{B}' \,|\!)^{w'} \cdot r) \\
& + (\sum c, w, r, w', v \mid [_z\, c{\downarrow_{w_{\mathfrak{c}}}}{\uparrow^{w'}}\, r\,]^{v} \in (\mathcal{B}')^2 : c \cdot (\!|_{w_{\mathfrak{c}}}\, \mathcal{B}' \,|\!)^{w'} \cdot r \cdot (\!|_v\, \mathcal{B}' \,|\!)^{z'}) \\
= & \qquad \{\!\{ \text{ Equation (3.22) } \}\!\} \\
& (\!|_z\, \mathcal{B}' \,|\!)^{z'}
\end{aligned}$$

Using (B.7), it is simple to prove that

$$\vdash \underset{x}{(\!|}\, \mathcal{B} \,\overset{y}{|\!)} = \underset{x_{\mathfrak{c}}}{(\!|}\, \mathcal{B}' \,\overset{y}{|\!)} \quad . \tag{B.8}$$

For the case $\geqslant$, use (3.37) with the following function $g : V' \to V$ defined by $g(z) := z$ and $g(z_{\mathfrak{c}}) := z$ for all $z \in V$. This works since the list $\widehat{g}(\mathcal{B}')$ shrinked with idempotency of blocks and swapping of blocks gives exactly $\mathcal{B}$. For the case $\leqslant$, by (B.7), it suffices to prove that

$$\vdash \underset{x}{(\!|}\, \mathcal{B} \,\overset{y}{|\!)} \leqslant \underset{x}{(\!|}\, \mathcal{B}' \,\overset{y}{|\!)} \quad .$$

Use (3.5) with $s_{(u,u')} := (\!|_u \mathcal{B}' |\!)^{u'}$ for all $u, u' \in V$. It suffices to show that, for all $u, u' \in V$,

$$(\wedge\, m \mid [\!\underset{u}{[}\, m \,]\!\overset{u'}{]} \in \mathcal{B}^1 : m \leqslant (\!|\underset{u}{\mathcal{B}'}|\!)\overset{u'}{}) \ , \tag{B.9}$$

$$(\wedge\, c, z, r, w \mid [\!\underset{u}{[}\, c{\downarrow_z}\overset{w}{\uparrow} r \,]\!\overset{u'}{]} \in \mathcal{B}^2 : c \cdot (\!|\underset{z}{\mathcal{B}'}|\!)\overset{w}{} \cdot r \leqslant (\!|\underset{u}{\mathcal{B}'}|\!)\overset{u'}{}) \ , \tag{B.10}$$

$$(\wedge\, v \mid v \in V : (\!|\underset{u}{\mathcal{B}'}|\!)\overset{v}{} \cdot (\!|\underset{v}{\mathcal{B}'}|\!)\overset{u'}{} \leqslant (\!|\underset{u}{\mathcal{B}'}|\!)\overset{u'}{}) \ . \tag{B.11}$$

Inequations (B.9) are proved easily by condition a of the definition of $\mathcal{B}'$ stating that $[\!|_u m |\!]^{u'} \in \mathcal{B}^1 \Leftrightarrow [\!|_u m |\!]^{u'} \in (\mathcal{B}')^1$ and axiom (3.2). Inequations (B.11) are proved easily with axiom (3.4).

For inequations (B.10),

$$\begin{array}{ll} & (\wedge\, c, z, r, w \mid [\!|_u c{\downarrow_z}{\uparrow^w} r |\!]^{u'} \in \mathcal{B}^2 : c \cdot (\!|_z \mathcal{B}' |\!)^w \cdot r \leqslant (\!|_u \mathcal{B}' |\!)^{u'}) \\ \leftrightarrow & \qquad \{\!\{ \text{ Definition of } \mathcal{B}', \text{ condition b: } [\!|_u c{\downarrow_z}{\uparrow^w} r |\!]^{u'} \in \mathcal{B}^2 \Leftrightarrow [\!|_u c{\downarrow_{z_\mathfrak{c}}}{\uparrow^w} r |\!]^{u'} \in (\mathcal{B}')^2 \ \}\!\} \\ & (\wedge\, c, z, r, w \mid [\!|_u c{\downarrow_{z_\mathfrak{c}}}{\uparrow^w} r |\!]^{u'} \in (\mathcal{B}')^2 : c \cdot (\!|_z \mathcal{B}' |\!)^w \cdot r \leqslant (\!|_u \mathcal{B}' |\!)^{u'}) \\ \leftrightarrow & \qquad \{\!\{ \text{ Equation (B.7) } \}\!\} \\ & (\wedge\, c, z, r, w \mid [\!|_u c{\downarrow_{z_\mathfrak{c}}}{\uparrow^w} r |\!]^{u'} \in (\mathcal{B}')^2 : c \cdot (\!|_{z_\mathfrak{c}} \mathcal{B}' |\!)^w \cdot r \leqslant (\!|_u \mathcal{B}' |\!)^{u'}) \\ & \qquad\qquad -\!\!-\ (\ \text{Axiom (3.3)} \ .\ ) \end{array}$$

**Transformation 2: Add Self-Looping Unary Blocks Containing 1 For All Labels**

The second transformation of the expression $(\!|_x \mathcal{B} |\!)^y$ is to simply add unary blocks of the form $[\!|_z 1 |\!]^z$ for every $z \in V'$. This is possible without "modifying the language" because of the previous cloning step. The previous step ensures that $x_\mathfrak{c} \neq y$ and, for every $[\!|_z c{\downarrow_w}{\uparrow^{w'}} r |\!]^{z'} \in (\mathcal{B}')^2$, $w \neq w'$ since $w \in \{u_\mathfrak{c} \mid u \in V\}$ and $w' \in V$.

Define $\mathcal{D}$ to be a finite list containing one, and only one, unary block $[\!|_z 1 |\!]^z$ for all $z \in V'$. We prove that

$$\vdash (\!|\underset{x_\mathfrak{c}}{\mathcal{B}'}|\!)\overset{y}{} = (\!|\underset{x_\mathfrak{c}}{\mathcal{B}', \mathcal{D}}|\!)\overset{y}{} \ . \tag{B.12}$$

The case $\leqslant$ is direct by (3.26). For the case $\geqslant$, use (3.5) with

$$s_{(u,u')} := \begin{cases} (\!|_u \mathcal{B}' |\!)^{u'} & \text{if } u \neq u', \\ 1 + (\!|_u \mathcal{B}' |\!)^{u'} & \text{otherwise}, \end{cases}$$

for all $u, u' \in V'$. It suffices to show that, for all $u, u' \in V'$ such that $u \neq u'$,

$$(\wedge\, m \mid [\!|_u m |\!]^{u'} \in (\mathcal{B}', \mathcal{D})^1 : m \leqslant (\!|_u \mathcal{B}' |\!)^{u'}) \ , \tag{B.13}$$

$$(\wedge\, m \mid [\!|_u m |\!]^{u} \in (\mathcal{B}', \mathcal{D})^1 : m \leqslant 1 + (\!|_u \mathcal{B}' |\!)^{u}) \ , \tag{B.14}$$

$$(\wedge\, c, z, r, w \mid [\!|_u c \downarrow_{z_c}^{w} \uparrow r |\!]^{u'} \in (\mathcal{B}', \mathcal{D})^2 : c \cdot (\!|_{z_c} \mathcal{B}' |\!)^{w} \cdot r \leqslant (\!|_u \mathcal{B}' |\!)^{u'}) \ , \tag{B.15}$$

$$(\wedge\, c, z, r, w \mid [\!|_u c \downarrow_{z_c}^{w} \uparrow r |\!]^{u} \in (\mathcal{B}', \mathcal{D})^2 : c \cdot (\!|_{z_c} \mathcal{B}' |\!)^{w} \cdot r \leqslant 1 + (\!|_u \mathcal{B}' |\!)^{u}) \ , \tag{B.16}$$

$$(\wedge\, v \mid v \in V' \wedge v \neq u \wedge v \neq u' : (\!|_u \mathcal{B}' |\!)^{v} \cdot (\!|_v \mathcal{B}' |\!)^{u'} \leqslant (\!|_u \mathcal{B}' |\!)^{u'}) \ , \tag{B.17}$$

$$(1 + (\!|_u \mathcal{B}' |\!)^{u}) \cdot (\!|_u \mathcal{B}' |\!)^{u'} \leqslant (\!|_u \mathcal{B}' |\!)^{u'} \ , \tag{B.18}$$

$$(\!|_u \mathcal{B}' |\!)^{u'} \cdot (1 + (\!|_{u'} \mathcal{B}' |\!)^{u'}) \leqslant (\!|_u \mathcal{B}' |\!)^{u'} \ , \tag{B.19}$$

$$(1 + (\!|_u \mathcal{B}' |\!)^{u}) \cdot (1 + (\!|_u \mathcal{B}' |\!)^{u}) \leqslant 1 + (\!|_u \mathcal{B}' |\!)^{u} \ . \tag{B.20}$$

To ease the understanding of the proofs, the concept of self-looping unary block is used.

**Definition B.2** (Self-looping unary block)**.** A unary block $[\!|_u m |\!]^{u'}$ is called *self-looping* if $u = u'$. Otherwise, it is called *non-self-looping*. ♦

For (B.13), first note that $\mathcal{D}$ does not contain a non-self-looping unary block. So, for every $m \in \Sigma_{\mathbf{i}} \cup \{0, 1\}$ and $u, u' \in V'$ such that $u \neq u'$,

$$[\!|_u m |\!]^{u'} \in (\mathcal{B}', \mathcal{D})^1 \Leftrightarrow [\!|_u m |\!]^{u'} \in (\mathcal{B}')^1 \ .$$

Then, to prove (B.13), it suffices to prove that, for every $m \in \Sigma_{\mathbf{i}} \cup \{0, 1\}$ such that $[\!|_u m |\!]^{u'} \in (\mathcal{B}')^1$,

$$m \leqslant (\!|_u \mathcal{B}' |\!)^{u'} \ .$$

This is direct by axiom (3.2).

For (B.14), first note that $\mathcal{D}$ contain one and only one self-looping unary block $[\!]_u\, 1\, [\!]^u$. So, to prove (B.14), it suffices to prove

$$1 \leqslant 1 + \underset{u}{(\!|}\, \mathcal{B}' \,\underset{u}{|\!)} \qquad \wedge \qquad (\wedge\, m \mid \underset{u}{[\!|}\, m \,\underset{u}{|\!]} \in (\mathcal{B}')^1 : m \leqslant 1 + \underset{u}{(\!|}\, \mathcal{B}' \,\underset{u}{|\!)}) \ .$$

The first formula is trivial by Kleene algebra. For the second formula, note that, by Kleene algebra, it suffices to prove that

$$(\wedge\, m \mid \underset{u}{[\!|}\, m \,\underset{u}{|\!]} \in (\mathcal{B}')^1 : m \leqslant \underset{u}{(\!|}\, \mathcal{B}' \,\underset{u}{|\!)}) \ .$$

This is direct by axiom (3.2).

For (B.15), first note that $\mathcal{D}$ does not contain a binary block. So, for every $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$, $z \in V$ and $w \in V'$,

$$\underset{u}{[\!|}\, c \underset{z_{\mathbf{c}}}{\downarrow} \overset{w}{\uparrow} r \,\overset{u'}{|\!]} \in (\mathcal{B}', \mathcal{D})^2 \Leftrightarrow \underset{u}{[\!|}\, c \underset{z_{\mathbf{c}}}{\downarrow} \overset{w}{\uparrow} r \,\overset{u'}{|\!]} \in (\mathcal{B}')^2 \ .$$

Then, to prove (B.15), it suffices to prove that, for every $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$, $z \in V$ and $w \in V'$ such that $[\!]_u\, c \downarrow_{z_{\mathbf{c}}} \uparrow^w r\, [\!]^{u'} \in (\mathcal{B}')^2$,

$$c \cdot \underset{z_{\mathbf{c}}}{(\!|}\, \mathcal{B}' \,\overset{w}{|\!)} \cdot r \leqslant \underset{u}{(\!|}\, \mathcal{B}' \,\overset{u'}{|\!)} \ .$$

This is direct by axiom (3.3).

For (B.16), first note that, by Kleene algebra, it suffices to prove that

$$(\wedge\, c, z, r, w \mid \underset{u}{[\!|}\, c \underset{z_{\mathbf{c}}}{\downarrow} \overset{w}{\uparrow} r \,\overset{u}{|\!]} \in (\mathcal{B}', \mathcal{D})^2 : c \cdot \underset{z_{\mathbf{c}}}{(\!|}\, \mathcal{B}' \,\overset{w}{|\!)} \cdot r \leqslant \underset{u}{(\!|}\, \mathcal{B}' \,\overset{u}{|\!)}) \ . \tag{B.21}$$

The proof of (B.21) is similar to the proof of (B.15).

Inequations (B.17) are proved easily with axiom (3.4).

For (B.18),

$$\begin{array}{ll} & (1 + (\!|_u\, \mathcal{B}' \,|\!)^u) \cdot (\!|_u\, \mathcal{B}' \,|\!)^{u'} \\ = & \qquad \{\!\{ \text{ Distributivity of } \cdot \text{ over } + \text{ \& Identity of } \cdot \ \}\!\} \\ & (\!|_u\, \mathcal{B}' \,|\!)^{u'} + (\!|_u\, \mathcal{B}' \,|\!)^u \cdot (\!|_u\, \mathcal{B}' \,|\!)^{u'} \\ = & \qquad \{\!\{ \text{ Axiom (3.4) and definition of } \leqslant \ \}\!\} \\ & (\!|_u\, \mathcal{B}' \,|\!)^{u'} \ . \end{array}$$

The proof of (B.19) is similar to the proof of (B.18).

For (B.20),

$$
\begin{aligned}
& (1 + (\!|_u \mathcal{B}' |\!)^u) \cdot (1 + (\!|_u \mathcal{B}' |\!)^u) \\
= & \quad \{\!\{ \text{ Distributivity of } \cdot \text{ over } + \ \}\!\} \\
& 1 \cdot 1 + 1 \cdot (\!|_u \mathcal{B}' |\!)^u + (\!|_u \mathcal{B}' |\!)^u \cdot 1 + (\!|_u \mathcal{B}' |\!)^u \cdot (\!|_u \mathcal{B}' |\!)^u \\
= & \quad \{\!\{ \text{ Identity of } \cdot \text{ \& Idempotency of } + \ \}\!\} \\
& 1 + (\!|_u \mathcal{B}' |\!)^u + (\!|_u \mathcal{B}' |\!)^u \cdot (\!|_u \mathcal{B}' |\!)^u \\
= & \quad \{\!\{ \text{ Axiom (3.4) and definition of } \leqslant \ \}\!\} \\
& 1 + (\!|_u \mathcal{B}' |\!)^u \ .
\end{aligned}
$$

**Transformation 3: Remove All Unary Blocks Containing** 0

The third transformation of the expression $(\!|_x \mathcal{B} |\!)^y$ is simply to remove unary blocks of the form $[_z\, 0\,]^{z'}$ for every $z, z' \in V'$. Let $\mathcal{B}''$ be the list $\mathcal{B}'$ in which every unary block of the form $[_z\, 0\,]^{z'}$ for every $z, z' \in V'$ is removed. By (3.36),

$$\vdash (\!|\, \mathcal{B}', \mathcal{D} \,|\!)^{y}_{x_{\mathfrak{c}}} = (\!|\, \mathcal{B}'', \mathcal{D} \,|\!)^{y}_{x_{\mathfrak{c}}} \ . \tag{B.22}$$

Note that it is always possible to remove such a block using (3.36) since there is at least one block in $\mathcal{D}$.

**Transformation 4: Remove All Non-Self-Looping Unary Blocks Containing** 1

The fourth transformation of the expression $(\!|_x \mathcal{B} |\!)^y$ is to get rid of unary blocks of the form $[_z\, 1\,]^{z'}$ such that $z \neq z'$. This is done by simulating reflexive transitive closure over unary blocks of the form $[_z\, 1\,]^{z'}$ for every $z, z' \in V'$.

Define the function $\mathtt{1t}^n_{\mathcal{B}'',\mathcal{D}} : V' \to 2^{V'}$ (relative to the list of blocks $\mathcal{B}'', \mathcal{D}$) that calculates the set of labels accessible from a received label by going "backward" in the list $\mathcal{B}'', \mathcal{D}$ through exactly $n$ blocks of the form $[_z\, 1\,]^{z'}$. So, for all labels $z \in V'$, $\mathtt{1t}^n_{\mathcal{B}'',\mathcal{D}}(z)$ is defined by induction on $n$:

- Base case $n = 0$: $\mathtt{1t}^0_{\mathcal{B}'',\mathcal{D}}(z) := \{z\}$.

- Induction case $n \geqslant 0$:

$$\mathsf{1t}^{n+1}_{\mathcal{B}'',\mathcal{D}}(z) := \{z' \in V' \mid (\exists\, z'' \mid: \underset{z''}{[\,1\,]^{z}} \in (\mathcal{B}'',\mathcal{D})^1 \wedge z' \in \mathsf{1t}^{n}_{\mathcal{B}'',\mathcal{D}}(z''))\} \ .$$

Note that, for all $z \in V'$ and $n \geqslant 0$, every $z'$ in $\mathsf{1t}^{n}_{\mathcal{B}'',\mathcal{D}}(z)$ is also in $\mathsf{1t}^{n+1}_{\mathcal{B}'',\mathcal{D}}(z)$ (this is trivial since $[_z\, 1\,]^z$ is always in $\mathcal{D}^1$).

Now, define $\mathsf{1t}^{*}_{\mathcal{B}'',\mathcal{D}} : V' \to 2^{V'}$ for all $z \in V'$ by

$$\mathsf{1t}^{*}_{\mathcal{B}'',\mathcal{D}}(z) := (\cup\, n \mid n \in \mathbb{N} : \mathsf{1t}^{n}_{\mathcal{B}'',\mathcal{D}}(z)) \ .$$

Note that, although this union is infinite, the calculation of $\mathsf{1t}^{*}_{\mathcal{B}'',\mathcal{D}}$ always stabilizes in a polynomial time in the size of $\mathcal{B}'', \mathcal{D}$ and the number of labels in $V'$.

Define the finite list $\mathcal{B}'''$ by the following rules:

a. for $z \in V'$, the unary block $[_z\, 1\,]^z$ is in $(\mathcal{B}''')^1$;

b. for $a \in \Sigma_{\mathbf{i}}$ and $z, z' \in V'$, the unary block $[_z\, a\,]^{z'}$ is in $(\mathcal{B}''')^1$ if there exists $z'' \in V'$ such that $[_z\, a\,]^{z''}$ is in $(\mathcal{B}'',\mathcal{D})^1$ and $z'' \in \mathsf{1t}^{*}_{\mathcal{B}'',\mathcal{D}}(z')$;

c. for $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $z, z', w, w' \in V'$, the binary block $[_z\, c{\downarrow_w}{\uparrow^{w'}}\, r\,]^{z'}$ is in $(\mathcal{B}''')^2$ if there exists $z'', w'' \in V'$ such that $[_z\, c{\downarrow_{w''}}{\uparrow^{w'}}\, r\,]^{z''}$ is in $(\mathcal{B}'',\mathcal{D})^2$, $w'' \in \mathsf{1t}^{*}_{\mathcal{B}'',\mathcal{D}}(w)$ and $z'' \in \mathsf{1t}^{*}_{\mathcal{B}'',\mathcal{D}}(z')$.

Note that condition b is defined only for every $a \in \Sigma_{\mathbf{i}}$ and not for $a = 1$.

We now prove that

$$\vdash \underset{x_{\mathfrak{c}}}{(\!|\, \mathcal{B}'', \mathcal{D}\,|\!)^{y}} = (\sum x' \mid x_{\mathfrak{c}} \in \mathsf{1t}^{*}_{\mathcal{B}'',\mathcal{D}}(x') : \underset{x'}{(\!|\, \mathcal{B}'''\,|\!)^{y}}) \ . \tag{B.23}$$

**Proof of the case $\leqslant$ of (B.23):** We use axiom (3.14) with

$$s_{(u,u')} := (\sum u'' \mid u \in \mathsf{1t}^{*}_{\mathcal{B}'',\mathcal{D}}(u'') : \underset{u''}{(\!|\, \mathcal{B}'''\,|\!)^{u'}})$$

for all $u, u' \in V'$. It suffices to prove that, for all $u, u' \in V'$,

$$(\wedge\, m \mid [\!\underset{u}{[}\, m \,]\!\overset{u'}{]} \in (\mathcal{B}'', \mathcal{D})^1 : m \leqslant s_{(u,u')}) \ , \tag{B.24}$$

$$(\wedge\, m, v \mid [\!\underset{u}{[}\, m \,]\!\overset{v}{]} \in (\mathcal{B}'', \mathcal{D})^1 : m \cdot s_{(v,u')} \leqslant s_{(u,u')}) \ , \tag{B.25}$$

$$(\wedge\, c, z, r, w \mid [\!\underset{u}{[}\, c \underset{z}{\downarrow} \overset{w}{\uparrow} r \,]\!\overset{u'}{]} \in (\mathcal{B}'', \mathcal{D})^2 : c \cdot s_{(z,w)} \cdot r \leqslant s_{(u,u')}) \ , \tag{B.26}$$

$$(\wedge\, c, z, r, w, v \mid [\!\underset{u}{[}\, c \underset{z}{\downarrow} \overset{w}{\uparrow} r \,]\!\overset{v}{]} \in (\mathcal{B}'', \mathcal{D})^2 : c \cdot s_{(z,w)} \cdot r \cdot s_{(v,u')} \leqslant s_{(u,u')}) \ . \tag{B.27}$$

For (B.24), we suppose $[\!\underset{u}{[}\, m \,]\!\overset{u'}{]} \in (\mathcal{B}'', \mathcal{D})^1$ and we prove that

$$m \leqslant (\sum u'' \mid u \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(u'') : (\!|\underset{u''}{\mathcal{B}'''}|\!)^{u'}) \ .$$

Let us prove this by case analysis on $m$. The case $m = 0$ does not exist by the third transformation (see Section B.9.1).

For the case $m = 1$, since $[\!\underset{u}{[}\, 1 \,]\!\overset{u'}{]} \in (\mathcal{B}'', \mathcal{D})^1$, then $u \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(u')$. So, by Kleene algebra, it suffices to prove

$$1 \leqslant (\!|\underset{u'}{\mathcal{B}'''}|\!)^{u'} \ .$$

This is direct from condition a of the definition of $\mathcal{B}'''$ and axiom (3.2).

For the case $m \in \Sigma_{\mathbf{i}}$, by the definition of $\mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}$ stating that $u \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(u)$ and by Kleene algebra, it suffices to prove

$$m \leqslant (\!|\underset{u}{\mathcal{B}'''}|\!)^{u'} \ .$$

By the definition of $\mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}$ stating that $u' \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(u')$, and by condition b of the definition of $\mathcal{B}'''$,

$$[\!\underset{u}{[}\, m \,]\!\overset{u'}{]} \in (\mathcal{B}'', \mathcal{D})^1 \Rightarrow [\!\underset{u}{[}\, m \,]\!\overset{u'}{]} \in (\mathcal{B}''')^1 \ .$$

So, by axiom (3.2), $m \leqslant (\!|_u\, \mathcal{B}'''\, |\!)^{u'}$.

For (B.25), we suppose $[\!\underset{u}{[}\, m \,]\!\overset{v}{]} \in (\mathcal{B}'', \mathcal{D})^1$ and we prove that

$$m \cdot (\sum v' \mid v \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(v') : (\!|\underset{v'}{\mathcal{B}'''}|\!)^{u'}) \leqslant (\sum u'' \mid u \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(u'') : (\!|\underset{u''}{\mathcal{B}'''}|\!)^{u'}) \ .$$

By Kleene algebra (distributivity of $\cdot$ over $+$ and the law $p_1+p_2 \leqslant p_3 \;\leftrightarrow\; p_1 \leqslant p_3 \wedge p_2 \leqslant p_3$), it suffices to prove independently that, for all $v' \in V'$ such that $v \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(v')$,

$$m \cdot \underset{v'}{\overset{u'}{(\!|\, \mathcal{B}''' \,|\!)}} \leqslant (\sum u'' \mid u \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(u'') : \underset{u''}{\overset{u'}{(\!|\, \mathcal{B}''' \,|\!)}}) \enspace .$$

Let us prove this by case analysis on $m$. The case $m = 0$ does not exist by the third transformation (see Section B.9.1).

For the case $m = 1$, since $[\!_u\, 1 \,]\!^v \in (\mathcal{B}'', \mathcal{D})^1$ and $v \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(v')$, then, by the definition of $\mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}$, $u \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(v')$. So, by Kleene algebra, it suffices to prove

$$1 \cdot \underset{v'}{\overset{u'}{(\!|\, \mathcal{B}''' \,|\!)}} \leqslant \underset{v'}{\overset{u'}{(\!|\, \mathcal{B}''' \,|\!)}} \enspace .$$

This is trivial by identity of $\cdot$.

For the case $m \in \Sigma_{\mathbf{i}}$, by the definition of $\mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}$ stating that $u \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(u)$ and by Kleene algebra, it suffices to prove that

$$m \cdot \underset{v'}{\overset{u'}{(\!|\, \mathcal{B}''' \,|\!)}} \leqslant \underset{u}{\overset{u'}{(\!|\, \mathcal{B}''' \,|\!)}} \enspace .$$

Since $[\!_u\, m \,]\!^v \in (\mathcal{B}'', \mathcal{D})^1$ and $v \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(v')$, then, by condition b of the definition of $\mathcal{B}'''$, $[\!_u\, m \,]\!^{v'} \in (\mathcal{B}''')^1$. So, by axiom (3.2), $m \leqslant (\!|_u\, \mathcal{B}''' \,|\!)^{v'}$. Then, by Kleene algebra, it suffices to prove that

$$\underset{u}{\overset{v'}{(\!|\, \mathcal{B}''' \,|\!)}} \cdot \underset{v'}{\overset{u'}{(\!|\, \mathcal{B}''' \,|\!)}} \leqslant \underset{u}{\overset{u'}{(\!|\, \mathcal{B}''' \,|\!)}} \enspace .$$

This is direct from axiom (3.4).

For (B.26), by Kleene algebra (distributivity of $\cdot$ over $+$ and the law $p_1+p_2 \leqslant p_3 \;\leftrightarrow\; p_1 \leqslant p_3 \;\wedge\; p_2 \leqslant p_3$), it suffices to prove independently that, for all $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $z, z', w \in V'$ such that $[\!_u\, c{\downarrow_z}{\uparrow^w}\, r \,]\!^{u'} \in (\mathcal{B}'', \mathcal{D})^2$ and $z \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(z')$,

$$c \cdot \underset{z'}{\overset{w}{(\!|\, \mathcal{B}''' \,|\!)}} \cdot r \leqslant (\sum u'' \mid u \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(u'') : \underset{u''}{\overset{u'}{(\!|\, \mathcal{B}''' \,|\!)}}) \enspace .$$

By the definition of $\mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}$ stating that $u \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(u)$ and by Kleene algebra, it suffices to prove that

$$c \cdot \underset{z'}{\overset{w}{(\!|\, \mathcal{B}''' \,|\!)}} \cdot r \leqslant \underset{u}{\overset{u'}{(\!|\, \mathcal{B}''' \,|\!)}} \enspace . \tag{B.28}$$

Since $[\!_u\, c{\downarrow_z}{\uparrow^w}\, r \,]\!^{u'} \in (\mathcal{B}'', \mathcal{D})^2$ and $z \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(z')$ by hypotheses, and $u' \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(u')$ by the definition of $\mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}$, then, by condition c of the definition of $\mathcal{B}'''$, $[\!_u\, c{\downarrow_{z'}}{\uparrow^w}\, r \,]\!^{u'} \in (\mathcal{B}''')^2$. So, inequation (B.28) is proved by axiom (3.3).

For (B.27), by Kleene algebra (distributivity of $\cdot$ over $+$ and the law $p_1 + p_2 \leqslant p_3 \leftrightarrow p_1 \leqslant p_3 \ \wedge \ p_2 \leqslant p_3$), it suffices to prove independently that, for all $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $z, z', w, v, v' \in V'$ such that $\lfloor_u c \downarrow_z \uparrow^w r \rfloor^v \in (\mathcal{B}'', \mathcal{D})^2$, $z \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(z')$ and $v \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(v')$,

$$c \cdot \underset{z'}{\overset{w}{(\!|}} \mathcal{B}''' \overset{w}{|\!)} \cdot r \cdot \underset{v'}{\overset{u'}{(\!|}} \mathcal{B}''' \overset{u'}{|\!)} \leqslant (\sum u'' \mid u \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(u'') : \underset{u''}{\overset{u'}{(\!|}} \mathcal{B}''' \overset{u'}{|\!)}) \ .$$

By the definition of $\mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}$ stating that $u \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(u)$ and by Kleene algebra, it suffices to prove that

$$c \cdot \underset{z'}{\overset{w}{(\!|}} \mathcal{B}''' \overset{w}{|\!)} \cdot r \cdot \underset{v'}{\overset{u'}{(\!|}} \mathcal{B}''' \overset{u'}{|\!)} \leqslant \underset{u}{\overset{u'}{(\!|}} \mathcal{B}''' \overset{u'}{|\!)} \ . \tag{B.29}$$

Since $\lfloor_u c \downarrow_z \uparrow^w r \rfloor^v \in (\mathcal{B}'', \mathcal{D})^2$, $z \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(z')$ and $v \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(v')$ by hypotheses, then, by condition c of the definition of $\mathcal{B}'''$, $\lfloor_u c \downarrow_{z'} \uparrow^w r \rfloor^{v'} \in (\mathcal{B}''')^2$. So, by axiom (3.3),

$$c \cdot \underset{z'}{\overset{w}{(\!|}} \mathcal{B}''' \overset{w}{|\!)} \cdot r \leqslant \underset{u}{\overset{v'}{(\!|}} \mathcal{B}''' \overset{v'}{|\!)} \ .$$

Then, to prove inequation (B.29), by Kleene algebra, it suffices to prove that

$$\underset{u}{\overset{v'}{(\!|}} \mathcal{B}''' \overset{v'}{|\!)} \cdot \underset{v'}{\overset{u'}{(\!|}} \mathcal{B}''' \overset{u'}{|\!)} \leqslant \underset{u}{\overset{u'}{(\!|}} \mathcal{B}''' \overset{u'}{|\!)} \ .$$

This is direct from axiom (3.4).

**Proof of the case $\geqslant$ of (B.23):** First, note that the following results hold. For every $z, z', z'' \in V'$ such that $z \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(z')$,

$$\vdash 1 \leqslant \underset{z}{\overset{z'}{(\!|}} \mathcal{B}'', \mathcal{D} \overset{z'}{|\!)} \ , \tag{B.30}$$

$$\vdash \underset{z'}{\overset{z''}{(\!|}} \mathcal{B}'', \mathcal{D} \overset{z''}{|\!)} \leqslant \underset{z}{\overset{z''}{(\!|}} \mathcal{B}'', \mathcal{D} \overset{z''}{|\!)} \ . \tag{B.31}$$

We prove (B.30). By the hypothesis $z \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(z')$ and the definition of $\mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}$, there exists at least a $n \in \mathbb{N}$ such that $z \in \mathsf{1t}^n_{\mathcal{B}'',\mathcal{D}}(z')$. Let $k$ be that number. In the case that $k = 0$, then, by the definition of $\mathsf{1t}^0_{\mathcal{B}'',\mathcal{D}}$, $z = z'$ and so

$$\vdash 1 \leqslant \underset{z}{\overset{z}{(\!|}} \mathcal{B}'', \mathcal{D} \overset{z}{|\!)}$$

since $\lfloor_z 1 \rfloor^z \in \mathcal{D}^1$ and axiom (3.2). In the case that $k \geqslant 1$, take $b_1 b_2 \ldots b_k$ a correct travelling starting with $z$, ending with $z'$ in which each $b_i$ is a unary block of the form $\lfloor_{w_i} 1 \rfloor^{w_{i+1}}$ for $i \in \{1, 2, \ldots, k\}$ and $w_i, w_{i+1} \in V'$. Of course, $w_1 = z$ and $w_{k+1} = z'$. It is easy to see that such a travelling $b_1 b_2 \ldots b_k$ always exists by the fact that $z \in \mathsf{1t}^n_{\mathcal{B}'',\mathcal{D}}(z')$. So,

$$
\begin{aligned}
& 1 \\
= & \qquad \{\!\{ \text{ Identity of } \cdot,\ k-1 \text{ times } \}\!\} \\
& \underbrace{1 \cdot 1 \cdot \ \cdots \ \cdot 1}_{\text{total of } k \text{ subexpressions}} \\
\leqslant & \qquad \{\!\{ \text{ Since each } b_i \text{ is a unary block of the form } \lfloor_{w_i}\, 1\, \rfloor^{w_{i+1}} \text{ for } i \in \{1,2,\ldots,k\} \text{ and } \\
& \qquad\quad w_i, w_{i+1} \in V', \text{ then } 1 \leqslant (\!|_{w_i}\, \mathcal{B}'', \mathcal{D}\, |\!)^{w_{i+1}} \text{ by axiom (3.2). \& Monotonicity of } \\
& \qquad\quad \cdot \ \}\!\} \\
& \underbrace{(\!|_{w_1}\, \mathcal{B}'', \mathcal{D}\, |\!)^{w_2} \cdot (\!|_{w_2}\, \mathcal{B}'', \mathcal{D}\, |\!)^{w_3} \cdot \ \cdots \ \cdot (\!|_{w_k}\, \mathcal{B}'', \mathcal{D}\, |\!)^{w_{k+1}}}_{\text{total of } k \text{ subexpressions}} \\
\leqslant & \qquad \{\!\{ \text{ Axiom (3.4), } k-1 \text{ times \& Monotonicity of } \cdot \ \}\!\} \\
& (\!|_{w_1}\, \mathcal{B}'', \mathcal{D}\, |\!)^{w_{k+1}} \\
= & \qquad \{\!\{ \text{ By definition, } w_1 = z \text{ and } w_{k+1} = z' \ \}\!\} \\
& (\!|_{z}\, \mathcal{B}'', \mathcal{D}\, |\!)^{z'} \ .
\end{aligned}
$$

We now prove (B.31).

$$
\begin{aligned}
& (\!|_{z'}\, \mathcal{B}'', \mathcal{D}\, |\!)^{z''} \\
= & \qquad \{\!\{ \text{ Identity of } \cdot \ \}\!\} \\
& 1 \cdot (\!|_{z'}\, \mathcal{B}'', \mathcal{D}\, |\!)^{z''} \\
\leqslant & \qquad \{\!\{ \text{ Hypothesis: } z \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(z') \text{ \& Inequation (B.30) \& Monotonicity of } \cdot \ \}\!\} \\
& (\!|_{z}\, \mathcal{B}'', \mathcal{D}\, |\!)^{z'} \cdot (\!|_{z'}\, \mathcal{B}'', \mathcal{D}\, |\!)^{z''} \\
\leqslant & \qquad \{\!\{ \text{ Axiom (3.4) } \}\!\} \\
& (\!|_{z}\, \mathcal{B}'', \mathcal{D}\, |\!)^{z''}
\end{aligned}
$$

To prove the case $\geqslant$ of (B.23), we use Kleene algebra to note that $p_1 + p_2 \leqslant p_3 \leftrightarrow p_1 \leqslant p_3 \wedge p_2 \leqslant p_3$. So, it suffices to prove independently that

$$\vdash \underset{x'}{(\!|}\, \mathcal{B}'''\, \underset{}{|\!)}^{y} \leqslant \underset{x_{\mathfrak{c}}}{(\!|}\, \mathcal{B}'', \mathcal{D}\, \underset{}{|\!)}^{y}$$

for all $x' \in V'$ such that $x_{\mathfrak{c}} \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(x')$. Let $x''$ be such a $x'$. By (B.31) and transitivity of $\leqslant$, it suffices to prove that

$$\vdash \underset{x''}{(\!|}\, \mathcal{B}'''\, \underset{}{|\!)}^{y} \leqslant \underset{x''}{(\!|}\, \mathcal{B}'', \mathcal{D}\, \underset{}{|\!)}^{y} \ .$$

We use (3.5) with $s_{(u,u')} := (\!|_u \mathcal{B}'', \mathcal{D} |\!)^{u'}$ for all $u, u' \in V'$. So, it suffices to prove that, for all $u, u' \in V'$,

$$(\wedge\ m \mid [\!|_u m |\!]^{u'} \in (\mathcal{B}''')^1 : m \leqslant (\!|_u \mathcal{B}'', \mathcal{D} |\!)^{u'}) \ , \tag{B.32}$$

$$(\wedge\ c, z, r, w \mid [\!|_u c{\downarrow_z}{\uparrow^w} r |\!]^{u'} \in (\mathcal{B}''')^2 : c \cdot (\!|_z \mathcal{B}'', \mathcal{D} |\!)^{w} \cdot r \leqslant (\!|_u \mathcal{B}'', \mathcal{D} |\!)^{u'}) \ , \tag{B.33}$$

$$(\wedge\ v \mid v \in V' : (\!|_u \mathcal{B}'', \mathcal{D} |\!)^{v} \cdot (\!|_v \mathcal{B}'', \mathcal{D} |\!)^{u'} \leqslant (\!|_u \mathcal{B}'', \mathcal{D} |\!)^{u'}) \ . \tag{B.34}$$

For inequations (B.32), we suppose $[\!|_u m |\!]^{u'} \in (\mathcal{B}''')^1$ and we prove

$$m \leqslant (\!|_u \mathcal{B}'', \mathcal{D} |\!)^{u'} \ .$$

Let us prove this by case analysis on $m$. The case $m = 0$ does not exist by the definition of $\mathcal{B}'''$.

For the case $m = 1$, by condition a of the definition of $\mathcal{B}'''$, it follows that $u = u'$. By the definition of $\mathcal{D}$, $[\!|_u 1 |\!]^u \in (\mathcal{B}'', \mathcal{D})^1$. So, the result follows by axiom (3.2).

For the case $m \in \Sigma_{\mathbf{i}}$, by the hypothesis that $[\!|_u m |\!]^{u'} \in (\mathcal{B}''')^1$ and by condition b of the definition of $\mathcal{B}'''$, there exists $u'' \in V'$ such that $[\!|_u m |\!]^{u''} \in (\mathcal{B}'', \mathcal{D})^1$ and $u'' \in \mathsf{1t}^*_{\mathcal{B}'', \mathcal{D}}(u')$. So,

$$\begin{array}{ll}
 & m \\
= & \quad \{\!\{ \text{ Identity of } \cdot \ \}\!\} \\
 & m \cdot 1 \\
\leqslant & \quad \{\!\{ \text{ Hypothesis: } [\!|_u m |\!]^{u''} \in (\mathcal{B}'', \mathcal{D})^1 \ \&\ \text{Axiom (3.2)} \ \&\ \text{Monotonicity of } \cdot \ \}\!\} \\
 & (\!|_u \mathcal{B}'', \mathcal{D} |\!)^{u''} \cdot 1 \\
\leqslant & \quad \{\!\{ \text{ Hypothesis: } u'' \in \mathsf{1t}^*_{\mathcal{B}'', \mathcal{D}}(u') \ \&\ \text{Inequation (B.30)} \ \&\ \text{Monotonicity of } \cdot \ \}\!\} \\
 & (\!|_u \mathcal{B}'', \mathcal{D} |\!)^{u''} \cdot (\!|_{u''} \mathcal{B}'', \mathcal{D} |\!)^{u'} \\
\leqslant & \quad \{\!\{ \text{ Axiom (3.4) } \}\!\} \\
 & (\!|_u \mathcal{B}'', \mathcal{D} |\!)^{u'} \ .
\end{array}$$

For inequations (B.33), we suppose $[\!|_u c{\downarrow_z}{\uparrow^w} r |\!]^{u'} \in (\mathcal{B}''')^2$ and we prove

$$c \cdot (\!|_z \mathcal{B}'', \mathcal{D} |\!)^{w} \cdot r \leqslant (\!|_u \mathcal{B}'', \mathcal{D} |\!)^{u'} \ .$$

By the hypothesis that $\lfloor_u c{\downarrow_z}{\uparrow^w} r \rfloor^{u'} \in (\mathcal{B}''')^2$ and by condition c of the definition of $\mathcal{B}'''$, there exists $u'', z' \in V'$ such that $\lfloor_u c{\downarrow_{z'}}{\uparrow^w} r \rfloor^{u''} \in (\mathcal{B}'', \mathcal{D})^2$, $z' \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(z)$ and $u'' \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(u')$. So,

$$
\begin{array}{ll}
 & c \cdot (\!|_z \mathcal{B}'', \mathcal{D} |\!)^w \cdot r \\
= & \qquad \{\!\{ \text{ Identity of } \cdot \ \}\!\} \\
 & c \cdot 1 \cdot (\!|_z \mathcal{B}'', \mathcal{D} |\!)^w \cdot r \cdot 1 \\
\leqslant & \qquad \{\!\{ \text{ Hypotheses: } z' \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(z) \text{ and } u'' \in \mathsf{1t}^*_{\mathcal{B}'',\mathcal{D}}(u') \text{ \& Inequation (B.30) \& Monotonicity of } \cdot \ \}\!\} \\
 & c \cdot (\!|_{z'} \mathcal{B}'', \mathcal{D} |\!)^z \cdot (\!|_z \mathcal{B}'', \mathcal{D} |\!)^w \cdot r \cdot (\!|_{u''} \mathcal{B}'', \mathcal{D} |\!)^{u'} \\
\leqslant & \qquad \{\!\{ \text{ Axiom (3.4) \& Monotonicity of } \cdot \ \}\!\} \\
 & c \cdot (\!|_{z'} \mathcal{B}'', \mathcal{D} |\!)^w \cdot r \cdot (\!|_{u''} \mathcal{B}'', \mathcal{D} |\!)^{u'} \\
\leqslant & \qquad \{\!\{ \text{ Hypothesis: } \lfloor_u c{\downarrow_{z'}}{\uparrow^w} r \rfloor^{u''} \in (\mathcal{B}'', \mathcal{D})^2 \text{ \& Axiom (3.3) \& Monotonicity of } \cdot \ \}\!\} \\
 & (\!|_u \mathcal{B}'', \mathcal{D} |\!)^{u''} \cdot (\!|_{u''} \mathcal{B}'', \mathcal{D} |\!)^{u'} \\
\leqslant & \qquad \{\!\{ \text{ Axiom (3.4) } \}\!\} \\
 & (\!|_u \mathcal{B}'', \mathcal{D} |\!)^{u'} \ .
\end{array}
$$

Inequations (B.34) are direct from axiom (3.4). ■

# Appendix C

# Proof of the Elimination of $\varepsilon$-Transitions (Theorem 4.7)

This proof is done by two successive transformations. First, we get rid of transitions of the form $(s, \varepsilon, \bot; s', \bot)$ for all $s, s' \in S$ in Theorem C.1. Second, transitions of the form $(s, \varepsilon, d; s', \bot)$ for all $s, s' \in S$ and $d \in \Gamma$ are removed in Theorem C.2. Note that, in our equations between matrices, we sometimes take the liberty of having different entry notation for the matrices. However, the correspondence between the notations will always be clear from the context and, of course, the matrices will always be of the same size.

## C.1 Step 1: Elimination of $\varepsilon$-Transitions of the Form $(s, \varepsilon, \bot; s', \bot)$ for All $s, s' \in S$

**Theorem C.1** (Elimination of $\varepsilon$-transitions of the form $(s, \varepsilon, \bot; s', \bot)$)**.** *Let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be three disjoint finite sets such that at least one of them is nonempty. Let $\mathcal{A} := (S, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma \cup \{\bot\}, \delta, I, F)$ be a semi-visibly pushdown automaton. Let*

$$(\overrightarrow{I}, \mathbf{WM}, \mathbf{T}_{\mathbf{c}}, \mathbf{T}_{\bot}, \varepsilon_{\bot}, \varepsilon_{\not\bot}, \overrightarrow{F})$$

*be the algebraic encoding of $\mathcal{A}$. Then, there exists a* S-VPA *$\mathcal{A}' := (S', \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma' \cup \{\bot\}, \delta', I', F')$ having an algebraic encoding*

$$(\overrightarrow{I'}, \mathbf{WM}', \mathbf{T}'_{\mathbf{c}}, \mathbf{T}'_{\bot}, \varepsilon'_{\bot}, \varepsilon'_{\not\bot}, \overrightarrow{F'})$$

*such that $\delta'$ does not contain $\varepsilon$-transitions of the form $(s, \varepsilon, \bot; s', \bot)$ for all $s, s' \in S'$ and*

$$\begin{aligned} \vdash\ & \overrightarrow{I}^{t} \bullet (\mathbf{T}_{\bot} + \mathbf{WM} + \varepsilon_{\bot} + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{+} \bullet \varepsilon_{\not\bot})^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet \overrightarrow{F} \\ =\ & \overrightarrow{I'}^{t} \bullet (\mathbf{T}'_{\bot} + \mathbf{WM}' + \varepsilon'_{\bot} + (\mathbf{T}'_{\mathsf{c}} \bullet \mathbf{WM}')^{+} \bullet \varepsilon'_{\not\bot})^{\star} \bullet (\mathbf{T}'_{\mathsf{c}} \bullet \mathbf{WM}')^{\star} \bullet \overrightarrow{F'} \ . \end{aligned} \tag{C.1}$$

*Also, $\mathcal{A}'$ has at most twice the number of states of $\mathcal{A}$.*

*Proof.* Define the function $\varepsilon\mathsf{t}^{*}_{\mathcal{A}} : S \to 2^{S}$ by

$$\varepsilon\mathsf{t}^{*}_{\mathcal{A}}(s) := \{s' \in S \mid \vdash \varepsilon^{\star}_{\bot}[s', s] = 1\}$$

for all $s \in S$. This function simulates the reflexive transitive closure of transitions of the form $(t_1, \varepsilon, \bot; t_2, \bot)$, but reversed: an expression $s' \in \varepsilon\mathsf{t}^{*}_{\mathcal{A}}(s)$ means that it is possible to start from state $s'$ and reach state $s$ by using only $\varepsilon$-transitions of the form $(t_1, \varepsilon, \bot; t_2, \bot)$. Note that every entry $(s', s)$ of $\varepsilon^{\star}_{\bot}$ can be reduced to 1 or 0 by the definition of $\varepsilon_{\bot}$, the definition of $^{\star}$, zero of $\cdot$, identity of $\cdot$ and laws: $0^{*} = 1$ and $1^{*} = 1$.

The automaton has states of the form $s_{\bot}$ and $s_{\not\bot}$ for each $s \in S$. The idea is just to encode in a state an information stating if the top of the stack is the bottom-of-stack symbol (this is represented by $s_{\bot}$) or not (this is represented by $s_{\not\bot}$).

Define the semi-visibly pushdown automaton

$$\mathcal{A}' := (\{s_{\bot}, s_{\not\bot} \mid s \in S\}, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \{d_{\bot}, d_{\not\bot} \mid d \in \Gamma\} \cup \{\bot\}, \delta', I', \{f_{\bot}, f_{\not\bot} \mid f \in F\})$$

where $I' := \{s_{\bot} \mid (\exists\ s' \mid s' \in I : s' \in \varepsilon\mathsf{t}^{*}_{\mathcal{A}}(s))\}$, and

$$\begin{aligned} \delta' := {} & \{(s_{\bot}, a, \lambda; s'_{\bot}, \lambda) \mid a \in \Sigma_{\mathbf{i}} \wedge (\exists\ s'' \mid s'' \in \varepsilon\mathsf{t}^{*}_{\mathcal{A}}(s') : (s, a, \lambda; s'', \lambda) \in \delta)\} \\ & \cup\ \{(s_{\not\bot}, a, \lambda; s'_{\not\bot}, \lambda) \mid a \in \Sigma_{\mathbf{i}} \wedge (s, a, \lambda; s', \lambda) \in \delta\} \\ & \cup\ \{(s_{\bot}, c, \lambda; s'_{\not\bot}, d_{\bot}), (s_{\not\bot}, c, \lambda; s'_{\not\bot}, d_{\not\bot}) \mid c \in \Sigma_{\mathbf{c}} \wedge (s, c, \lambda; s', d) \in \delta\} \\ & \cup\ \{(s_{\not\bot}, r, d_{\bot}; s'_{\bot}, \lambda) \mid r \in \Sigma_{\mathbf{r}} \wedge (\exists\ s'' \mid s'' \in \varepsilon\mathsf{t}^{*}_{\mathcal{A}}(s') : (s, r, d; s'', \lambda) \in \delta)\} \\ & \cup\ \{(s_{\not\bot}, r, d_{\not\bot}; s'_{\not\bot}, \lambda) \mid r \in \Sigma_{\mathbf{r}} \wedge (s, r, d; s', \lambda) \in \delta\} \\ & \cup\ \{(s_{\bot}, r, \bot; s'_{\bot}, \bot) \mid r \in \Sigma_{\mathbf{r}} \wedge (\exists\ s'' \mid s'' \in \varepsilon\mathsf{t}^{*}_{\mathcal{A}}(s') : (s, r, \bot; s'', \bot) \in \delta)\} \\ & \cup\ \{(s_{\not\bot}, \varepsilon, d_{\not\bot}; s'_{\bot}, \bot), (s_{\not\bot}, \varepsilon, d_{\bot}; s'_{\bot}, \bot) \mid (\exists\ s'' \mid s'' \in \varepsilon\mathsf{t}^{*}_{\mathcal{A}}(s') : (s, \varepsilon, d; s'', \bot) \in \delta)\} \ . \end{aligned}$$

The algebraic encoding of this automaton is

$$\left( \left[ \begin{array}{c} \overrightarrow{I} \bullet \varepsilon^{\star}_{\bot} \\ \hline \overrightarrow{0} \end{array} \right], \mathbf{WM}', \left[ \begin{array}{c|c} \mathbf{0} & \mathbf{T}_{\mathsf{c}} \\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}} \end{array} \right], \left[ \begin{array}{c|c} \mathbf{T}_{\bot} \bullet \varepsilon^{\star}_{\bot} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} \end{array} \right], \mathbf{0}, \left[ \begin{array}{c|c} \mathbf{0} & \mathbf{0} \\ \hline \varepsilon_{\not\bot} \bullet \varepsilon^{\star}_{\bot} & \mathbf{0} \end{array} \right], \left[ \begin{array}{c} \overrightarrow{F} \\ \hline \overrightarrow{F} \end{array} \right] \right) \ .$$

First, two results are given. Using these results, the proof of (C.1) is presented. Then, the two results are proved.

To ease the proof of (C.1), recall that the matrix $\mathbf{WM}$ is defined with respect to a list of blocks $\mathcal{B}$ encoding the structure of the semi-visibly pushdown automaton (except for $\varepsilon$-transitions). We first define a list of blocks $\mathcal{C}$ over labels $S \cup \{s_{\mathfrak{c}} \mid s \in S\}$ (where each label $s_{\mathfrak{c}}$ is fresh) that is very similar to $\mathcal{B}$ but does not have unary blocks $\lfloor_s\, 1\,\rfloor^s$ for $s \in S$ (and binary blocks are a little different). In fact, the list of blocks $\mathcal{C}$ is defined by the following propositions:

(a) $\lfloor_{s_{\mathfrak{c}}}\, 1\,\rfloor^{s_{\mathfrak{c}}} \in \mathcal{C}^1$ for all $s \in S$;

(b) $\lfloor_s\, a\,\rfloor^{s'} \in \mathcal{C}^1$ and $\lfloor_{s_{\mathfrak{c}}}\, a\,\rfloor^{s'_{\mathfrak{c}}} \in \mathcal{C}^1$ for all $s, s' \in S$, $a \in \Sigma_{\mathbf{i}}$ and $\lfloor_s\, a\,\rfloor^{s'} \in \mathcal{B}^1$;

(c) $\lfloor_s\, c{\downarrow_{t_{\mathfrak{c}}}}{\uparrow^{t'_{\mathfrak{c}}}}\, r\,\rfloor^{s'} \in \mathcal{C}^2$ and $\lfloor_{s_{\mathfrak{c}}}\, c{\downarrow_{t_{\mathfrak{c}}}}{\uparrow^{t'_{\mathfrak{c}}}}\, r\,\rfloor^{s'_{\mathfrak{c}}} \in \mathcal{C}^2$ for all $s, s', t, t' \in S$, $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $\lfloor_s\, c{\downarrow_t}{\uparrow^{t'}}\, r\,\rfloor^{s'} \in \mathcal{B}^2$.

Note that

$$\vdash \mathbf{WM} = \mathbf{I} + \mathbf{WM}_{\mathfrak{c}} \tag{C.2}$$

where $\mathbf{WM}_{\mathfrak{c}}$ is a matrix of size $|S| \times |S|$ in which each entry $(s, s') \in S \times S$ is exactly $(\!|_s\, \mathcal{C}\, |\!)^{s'}$.

Also, to ease the proof of (C.1), note that

$$\vdash \mathbf{WM}' = \left[\begin{array}{c|c} (\mathbf{WM}_{\mathfrak{c}} \bullet \varepsilon_\perp^\star)^\star & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM} \end{array}\right] . \tag{C.3}$$

Now, it is possible to prove (C.1).

$$\begin{aligned} &\left[\begin{array}{c|c} \overrightarrow{I}^t \bullet \varepsilon_\perp^\star & \overrightarrow{0} \end{array}\right] \bullet \left( \left[\begin{array}{c|c} \mathbf{T}_\perp \bullet \varepsilon_\perp^\star & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} \end{array}\right] + \mathbf{WM}' + \mathbf{0} \right. \\ &\qquad \left. + \left( \left[\begin{array}{c|c} \mathbf{0} & \mathbf{T}_{\mathsf{c}} \\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}} \end{array}\right] \bullet \mathbf{WM}' \right)^+ \bullet \left[\begin{array}{c|c} \mathbf{0} & \mathbf{0} \\ \hline \varepsilon_{\not\perp} \bullet \varepsilon_\perp^\star & \mathbf{0} \end{array}\right] \right)^\star \\ &\bullet \left( \left[\begin{array}{c|c} \mathbf{0} & \mathbf{T}_{\mathsf{c}} \\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}} \end{array}\right] \bullet \mathbf{WM}' \right)^\star \bullet \left[\begin{array}{c} \overrightarrow{F} \\ \hline \overrightarrow{F} \end{array}\right] \\ = \quad & \{\!\!\{ \text{ Identity of } + \text{ \& Equation (C.3) } \}\!\!\} \end{aligned}$$

$$
\begin{array}{ll}
 & \left[\begin{array}{c|c} \overrightarrow{I}^{\,t} \bullet \varepsilon_{\perp}^{\star} & \overrightarrow{0} \end{array}\right] \bullet \left( \left[\begin{array}{c|c} \mathbf{T}_{\perp} \bullet \varepsilon_{\perp}^{\star} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} \end{array}\right] + \left[\begin{array}{c|c} (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_{\perp}^{\star})^{\star} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM} \end{array}\right] \right. \\
 & \qquad \left. + \left( \left[\begin{array}{c|c} \mathbf{0} & \mathbf{T}_{\mathsf{c}} \\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}} \end{array}\right] \bullet \left[\begin{array}{c|c} (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_{\perp}^{\star})^{\star} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM} \end{array}\right] \right)^{+} \bullet \left[\begin{array}{c|c} \mathbf{0} & \mathbf{0} \\ \hline \varepsilon_{\not\perp} \bullet \varepsilon_{\perp}^{\star} & \mathbf{0} \end{array}\right] \right)^{\star} \\
 & \bullet \left( \left[\begin{array}{c|c} \mathbf{0} & \mathbf{T}_{\mathsf{c}} \\ \hline \mathbf{0} & \mathbf{T}_{\mathsf{c}} \end{array}\right] \bullet \left[\begin{array}{c|c} (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_{\perp}^{\star})^{\star} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM} \end{array}\right] \right)^{\star} \bullet \left[\begin{array}{c} \overrightarrow{F} \\ \hline \overrightarrow{F} \end{array}\right] \\
= & \qquad \{\!\{ \text{ Definition of } \bullet, {}^{+} \text{ and } {}^{\star} \text{ \& Kleene algebra } \}\!\} \\
 & \left[\begin{array}{c|c} \overrightarrow{I}^{\,t} \bullet \varepsilon_{\perp}^{\star} & \overrightarrow{0} \end{array}\right] \bullet \left( \left[\begin{array}{c|c} \mathbf{T}_{\perp} \bullet \varepsilon_{\perp}^{\star} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} \end{array}\right] + \left[\begin{array}{c|c} (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_{\perp}^{\star})^{\star} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM} \end{array}\right] \right. \\
 & \qquad \left. + \left[\begin{array}{c|c} \mathbf{0} & (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{+} \\ \hline \mathbf{0} & (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{+} \end{array}\right] \bullet \left[\begin{array}{c|c} \mathbf{0} & \mathbf{0} \\ \hline \varepsilon_{\not\perp} \bullet \varepsilon_{\perp}^{\star} & \mathbf{0} \end{array}\right] \right)^{\star} \\
 & \bullet \left[\begin{array}{c|c} \mathbf{I} & (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{+} \\ \hline \mathbf{0} & (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \end{array}\right] \bullet \left[\begin{array}{c} \overrightarrow{F} \\ \hline \overrightarrow{F} \end{array}\right] \\
= & \qquad \{\!\{ \text{ Definition of } \bullet \text{ and } + \}\!\} \\
 & \left[\begin{array}{c|c} \overrightarrow{I}^{\,t} \bullet \varepsilon_{\perp}^{\star} & \overrightarrow{0} \end{array}\right] \bullet \left[\begin{array}{c|c} \mathbf{T}_{\perp} \bullet \varepsilon_{\perp}^{\star} + (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_{\perp}^{\star})^{\star} + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{+} \bullet \varepsilon_{\not\perp} \bullet \varepsilon_{\perp}^{\star} & \mathbf{0} \\ \hline (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{+} \bullet \varepsilon_{\not\perp} \bullet \varepsilon_{\perp}^{\star} & \mathbf{WM} \end{array}\right]^{\star} \\
 & \bullet \left[\begin{array}{c|c} \mathbf{I} & (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{+} \\ \hline \mathbf{0} & (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \end{array}\right] \bullet \left[\begin{array}{c} \overrightarrow{F} \\ \hline \overrightarrow{F} \end{array}\right] \\
= & \qquad \{\!\{ \text{ Definition of } {}^{\star} \text{ \& Kleene algebra } \}\!\} \\
 & \left[\begin{array}{c|c} \overrightarrow{I}^{\,t} \bullet \varepsilon_{\perp}^{\star} & \overrightarrow{0} \end{array}\right] \bullet \left[\begin{array}{c|c} \begin{array}{c} (\mathbf{T}_{\perp} \bullet \varepsilon_{\perp}^{\star} + (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_{\perp}^{\star})^{\star} \\ + \ (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{+} \bullet \varepsilon_{\not\perp} \bullet \varepsilon_{\perp}^{\star})^{\star} \end{array} & \mathbf{0} \\ \hline \begin{array}{c} \mathbf{WM}^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{+} \bullet \varepsilon_{\not\perp} \bullet \varepsilon_{\perp}^{\star} \bullet \\ (\mathbf{T}_{\perp} \bullet \varepsilon_{\perp}^{\star} + (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_{\perp}^{\star})^{\star} \\ + \ (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{+} \bullet \varepsilon_{\not\perp} \bullet \varepsilon_{\perp}^{\star})^{\star} \end{array} & \mathbf{WM}^{\star} \end{array}\right] \\
 & \bullet \left[\begin{array}{c|c} \mathbf{I} & (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{+} \\ \hline \mathbf{0} & (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \end{array}\right] \bullet \left[\begin{array}{c} \overrightarrow{F} \\ \hline \overrightarrow{F} \end{array}\right] \\
= & \qquad \{\!\{ \text{ Definition of } \bullet \text{ \& Kleene algebra } \}\!\} \\
 & \left[\begin{array}{c|c} \overrightarrow{I}^{\,t} \bullet \varepsilon_{\perp}^{\star} \bullet (\mathbf{T}_{\perp} \bullet \varepsilon_{\perp}^{\star} + (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_{\perp}^{\star})^{\star} + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{+} \bullet \varepsilon_{\not\perp} \bullet \varepsilon_{\perp}^{\star})^{\star} & \overrightarrow{0} \end{array}\right] \\
 & \bullet \left[\begin{array}{c} (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet \overrightarrow{F} \\ \hline (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet \overrightarrow{F} \end{array}\right] \\
= & \qquad \{\!\{ \text{ Definition of } \bullet \text{ \& Kleene algebra } \}\!\} \\
 & \overrightarrow{I}^{\,t} \bullet \varepsilon_{\perp}^{\star} \bullet (\mathbf{T}_{\perp} \bullet \varepsilon_{\perp}^{\star} + (\mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_{\perp}^{\star})^{\star} + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{+} \bullet \varepsilon_{\not\perp} \bullet \varepsilon_{\perp}^{\star})^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet \overrightarrow{F} \\
= & \qquad \{\!\{ \text{ Kleene algebra: } (p^{\star} + q)^{*} = (p + q)^{*} \}\!\} \\
 & \overrightarrow{I}^{\,t} \bullet \varepsilon_{\perp}^{\star} \bullet (\mathbf{T}_{\perp} \bullet \varepsilon_{\perp}^{\star} + \mathbf{WM}_{\mathsf{c}} \bullet \varepsilon_{\perp}^{\star} + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{+} \bullet \varepsilon_{\not\perp} \bullet \varepsilon_{\perp}^{\star})^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet \overrightarrow{F} \\
= & \qquad \{\!\{ \text{ Distributivity of } \cdot \text{ over } + \}\!\} \\
 & \overrightarrow{I}^{\,t} \bullet \varepsilon_{\perp}^{\star} \bullet ((\mathbf{T}_{\perp} + \mathbf{WM}_{\mathsf{c}} + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{+} \bullet \varepsilon_{\not\perp}) \bullet \varepsilon_{\perp}^{\star})^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet \overrightarrow{F}
\end{array}
$$

$$
\begin{array}{ll}
= & \{\!\{ \text{ Denesting rule } \}\!\} \\
& \overrightarrow{I}^{\,t} \bullet (\mathbf{T}_\perp + \mathbf{WM}_\mathfrak{c} + \varepsilon_\perp + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star \bullet (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \\
= & \{\!\{ \text{ Kleene algebra: } p^* = (1+p)^* \}\!\} \\
& \overrightarrow{I}^{\,t} \bullet (\mathbf{T}_\perp + \mathbf{I} + \mathbf{WM}_\mathfrak{c} + \varepsilon_\perp + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star \bullet (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \\
= & \{\!\{ \text{ Equation (C.2) } \}\!\} \\
& \overrightarrow{I}^{\,t} \bullet (\mathbf{T}_\perp + \mathbf{WM} + \varepsilon_\perp + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star \bullet (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \qquad \blacksquare
\end{array}
$$

### C.1.1 Proof of (C.2)

It is routine to prove (C.2).

**Proof of the case $\geqslant$ of (C.2):** It is trivial since $\mathbf{I} \leqslant \mathbf{WM}$ by Lemma 4.5, property i, and $\mathbf{WM}_\mathfrak{c} \leqslant \mathbf{WM}$ by (3.37) using the substitution function $g : S \cup \{s_\mathfrak{c} \mid s \in S\} \to S$ defined by $g(s) := s$ and $g(s_\mathfrak{c}) := s$, and noting that $\hat{g}(\mathcal{C})$ shrinked with idempotency of blocks and swapping of blocks gives exactly $\mathcal{B}$.

**Proof of the case $\leqslant$ of (C.2):** First note that for all $s, s' \in S$,

$$\vdash \underset{s}{(\!|} \mathcal{C} \overset{s'}{|\!)} \leqslant \underset{s_\mathfrak{c}}{(\!|} \mathcal{C} \overset{s'_\mathfrak{c}}{|\!)} \tag{C.4}$$

by (3.37) using the substitution function $h : S \cup \{s_\mathfrak{c} \mid s \in S\} \to S \cup \{s_\mathfrak{c} \mid s \in S\}$ defined by $h(s) := s_\mathfrak{c}$ and $h(s_\mathfrak{c}) := s_\mathfrak{c}$, and noting that $\hat{h}(\mathcal{C})$ shrinked with idempotency of blocks and swapping of blocks gives a sublist of $\mathcal{C}$. So, inequation (C.4) follows by (3.26) and (3.30).

To prove the case $\leqslant$ of (C.2) we use (3.5) with

$$s_{(u,u')} := \begin{cases} \underset{u}{(\!|} \mathcal{C} |\!)^{u'} & \text{if } u \neq u', \\ 1 + \underset{u}{(\!|} \mathcal{C} |\!)^{u'} & \text{otherwise,} \end{cases}$$

for all $u, u' \in S$. It suffices to prove that, for all $u, u' \in S$ such that $u \neq u'$,

$$(\wedge\, m \mid \underset{u}{[} m \overset{u'}{]} \in \mathcal{B}^1 : m \leqslant \underset{u}{(\!|} \mathcal{C} \overset{u'}{|\!)}) \ , \tag{C.5}$$

$$(\wedge\, m \mid \underset{u}{[} m \overset{u}{]} \in \mathcal{B}^1 : m \leqslant 1 + \underset{u}{(\!|} \mathcal{C} \overset{u}{|\!)}) \ , \tag{C.6}$$

$$(\wedge\ c, z, r, w \mid [\,{}_{u}\, c{\downarrow_{z}}{\uparrow^{w}}\, r\,]^{u'} \in \mathcal{B}^2 \wedge z \neq w : c \cdot {}_{z}(\!|\mathcal{C}|\!)^{w} \cdot r \leqslant {}_{u}(\!|\mathcal{C}|\!)^{u'}) \ , \tag{C.7}$$

$$(\wedge\ c, z, r \mid [\,{}_{u}\, c{\downarrow_{z}}{\uparrow^{z}}\, r\,]^{u'} \in \mathcal{B}^2 : c \cdot (1 + {}_{z}(\!|\mathcal{C}|\!)^{z}) \cdot r \leqslant {}_{u}(\!|\mathcal{C}|\!)^{u'}) \ , \tag{C.8}$$

$$(\wedge\ c, z, r, w \mid [\,{}_{u}\, c{\downarrow_{z}}{\uparrow^{w}}\, r\,]^{u} \in \mathcal{B}^2 \wedge z \neq w : c \cdot {}_{z}(\!|\mathcal{C}|\!)^{w} \cdot r \leqslant 1 + {}_{u}(\!|\mathcal{C}|\!)^{u}) \ , \tag{C.9}$$

$$(\wedge\ c, z, r \mid [\,{}_{u}\, c{\downarrow_{z}}{\uparrow^{z}}\, r\,]^{u} \in \mathcal{B}^2 : c \cdot (1 + {}_{z}(\!|\mathcal{C}|\!)^{z}) \cdot r \leqslant 1 + {}_{u}(\!|\mathcal{C}|\!)^{u}) \ , \tag{C.10}$$

$$(\wedge\ v \mid v \in S \wedge v \neq u \wedge v \neq u' : {}_{u}(\!|\mathcal{C}|\!)^{v} \cdot {}_{v}(\!|\mathcal{C}|\!)^{u'} \leqslant {}_{u}(\!|\mathcal{C}|\!)^{u'}) \ , \tag{C.11}$$

$$(1 + {}_{u}(\!|\mathcal{C}|\!)^{u}) \cdot {}_{u}(\!|\mathcal{C}|\!)^{u'} \leqslant {}_{u}(\!|\mathcal{C}|\!)^{u'} \ , \tag{C.12}$$

$${}_{u}(\!|\mathcal{C}|\!)^{u'} \cdot (1 + {}_{u'}(\!|\mathcal{C}|\!)^{u'}) \leqslant {}_{u}(\!|\mathcal{C}|\!)^{u'} \ , \tag{C.13}$$

$$(1 + {}_{u}(\!|\mathcal{C}|\!)^{u}) \cdot (1 + {}_{u}(\!|\mathcal{C}|\!)^{u}) \leqslant 1 + {}_{u}(\!|\mathcal{C}|\!)^{u} \ . \tag{C.14}$$

For inequations (C.5), we suppose $[\,{}_{u}\, m\,]^{u'} \in \mathcal{B}^1$ and we prove

$$m \leqslant {}_{u}(\!|\mathcal{C}|\!)^{u'} \ .$$

Since $u \neq u'$ by hypothesis, then, by the definition of $\mathcal{B}$, $m \in \Sigma_{\mathbf{i}}$. So, by condition b of the definition of $\mathcal{C}$, $[\,{}_{u}\, m\,]^{u'} \in \mathcal{C}^1$. Then, the proof follows from axiom (3.2).

For inequations (C.6), we suppose $[\,{}_{u}\, m\,]^{u} \in \mathcal{B}^1$ and we prove

$$m \leqslant 1 + {}_{u}(\!|\mathcal{C}|\!)^{u} \ .$$

Let us prove this by case analysis on $m$. The case $m = 0$ does not exist by the definition of $\mathcal{B}$. The case $m = 1$ is trivial by Kleene algebra (after all, $1 \leqslant 1$). For the case $m \in \Sigma_{\mathbf{i}}$, by condition b of the definition of $\mathcal{C}$, $[\,{}_{u}\, m\,]^{u} \in \mathcal{C}^1$. Then, the proof follows from axiom (3.2).

For inequations (C.7), we suppose $[\,{}_{u}\, c{\downarrow_{z}}{\uparrow^{w}}\, r\,]^{u'} \in \mathcal{B}^2$ and $z \neq w$, and we prove

$$c \cdot {}_{z}(\!|\mathcal{C}|\!)^{w} \cdot r \leqslant {}_{u}(\!|\mathcal{C}|\!)^{u'} \ .$$

By (C.4) and Kleene algebra, it suffices to prove that

$$c \cdot {}_{z_{\mathfrak{c}}}(\!|\mathcal{C}|\!)^{w_{\mathfrak{c}}} \cdot r \leqslant {}_{u}(\!|\mathcal{C}|\!)^{u'} \ .$$

Since $\lfloor_u c\downarrow_z\uparrow^w r\rfloor^{u'} \in \mathcal{B}^2$ by hypothesis, then, by condition c of the definition of $\mathcal{C}$, $\lfloor_u c\downarrow_{z_\mathfrak{c}}\uparrow^{w_\mathfrak{c}} r\rfloor^{u'} \in \mathcal{C}^2$. So, the proof follows from axiom (3.3).

For inequations (C.8), we suppose $\lfloor_u c\downarrow_z\uparrow^z r\rfloor^{u'} \in \mathcal{B}^2$, and we prove

$$c\cdot(1+\underset{z}{(\!|}\mathcal{C}\underset{}{|\!)}^{z})\cdot r \leqslant \underset{u}{(\!|}\mathcal{C}\underset{}{|\!)}^{u'} \quad .$$

By (C.4) and Kleene algebra, it suffices to prove that

$$c\cdot(1+\underset{z_\mathfrak{c}}{(\!|}\mathcal{C}\underset{}{|\!)}^{z_\mathfrak{c}})\cdot r \leqslant \underset{u}{(\!|}\mathcal{C}\underset{}{|\!)}^{u'} \quad .$$

Since $\lfloor_{z_\mathfrak{c}} 1\rfloor^{z_\mathfrak{c}} \in \mathcal{C}^1$ by condition a of the definition of $\mathcal{C}$, then $1 \leqslant (\!|_{z_\mathfrak{c}}\,\mathcal{C}\,|\!)^{z_\mathfrak{c}}$ by axiom (3.2). So, by definition of $\leqslant$, it suffices to prove that

$$c\cdot\underset{z_\mathfrak{c}}{(\!|}\mathcal{C}\underset{}{|\!)}^{z_\mathfrak{c}}\cdot r \leqslant \underset{u}{(\!|}\mathcal{C}\underset{}{|\!)}^{u'} \quad .$$

Since $\lfloor_u c\downarrow_z\uparrow^z r\rfloor^{u'} \in \mathcal{B}^2$ by hypothesis, then, by condition c of the definition of $\mathcal{C}$, $\lfloor_u c\downarrow_{z_\mathfrak{c}}\uparrow^{z_\mathfrak{c}} r\rfloor^{u'} \in \mathcal{C}^2$. So, the proof follows from axiom (3.3).

For inequations (C.9), we suppose $\lfloor_u c\downarrow_z\uparrow^w r\rfloor^u \in \mathcal{B}^2$ and $z \neq w$, and we prove

$$c\cdot\underset{z}{(\!|}\mathcal{C}\underset{}{|\!)}^{w}\cdot r \leqslant 1+\underset{u}{(\!|}\mathcal{C}\underset{}{|\!)}^{u} \quad .$$

By Kleene algebra, it suffices to prove that

$$c\cdot\underset{z}{(\!|}\mathcal{C}\underset{}{|\!)}^{w}\cdot r \leqslant \underset{u}{(\!|}\mathcal{C}\underset{}{|\!)}^{u} \quad .$$

The proof of this result is similar to the proof of (C.7).

For inequations (C.10), we suppose $\lfloor_u c\downarrow_z\uparrow^z r\rfloor^u \in \mathcal{B}^2$, and we prove

$$c\cdot(1+\underset{z}{(\!|}\mathcal{C}\underset{}{|\!)}^{z})\cdot r \leqslant 1+\underset{u}{(\!|}\mathcal{C}\underset{}{|\!)}^{u} \quad .$$

By Kleene algebra, it suffices to prove that

$$c\cdot(1+\underset{z}{(\!|}\mathcal{C}\underset{}{|\!)}^{z})\cdot r \leqslant \underset{u}{(\!|}\mathcal{C}\underset{}{|\!)}^{u} \quad .$$

The proof of this result is similar to the proof of (C.8).

Inequations (C.11) are proved easily with axiom (3.4).

For (C.12),

$$
\begin{aligned}
& (1 + (\!|_u \mathcal{C} |\!)^u) \cdot (\!|_u \mathcal{C} |\!)^{u'} \\
= & \qquad \{\!\{ \text{ Distributivity of } \cdot \text{ over } + \text{ \& Identity of } \cdot \ \}\!\} \\
& (\!|_u \mathcal{C} |\!)^{u'} + (\!|_u \mathcal{C} |\!)^u \cdot (\!|_u \mathcal{C} |\!)^{u'} \\
= & \qquad \{\!\{ \text{ Axiom (3.4) and definition of } \leqslant \ \}\!\} \\
& (\!|_u \mathcal{C} |\!)^{u'} \ .
\end{aligned}
$$

The proof of (C.13) is similar to the proof of (C.12).

For (C.14),

$$
\begin{aligned}
& (1 + (\!|_u \mathcal{C} |\!)^u) \cdot (1 + (\!|_u \mathcal{C} |\!)^u) \\
= & \qquad \{\!\{ \text{ Distributivity of } \cdot \text{ over } + \ \}\!\} \\
& 1 \cdot 1 + 1 \cdot (\!|_u \mathcal{C} |\!)^u + (\!|_u \mathcal{C} |\!)^u \cdot 1 + (\!|_u \mathcal{C} |\!)^u \cdot (\!|_u \mathcal{C} |\!)^u \\
= & \qquad \{\!\{ \text{ Identity of } \cdot \text{ \& Idempotency of } + \ \}\!\} \\
& 1 + (\!|_u \mathcal{C} |\!)^u + (\!|_u \mathcal{C} |\!)^u \cdot (\!|_u \mathcal{C} |\!)^u \\
= & \qquad \{\!\{ \text{ Axiom (3.4) and definition of } \leqslant \ \}\!\} \\
& 1 + (\!|_u \mathcal{C} |\!)^u \ .
\end{aligned}
$$

### C.1.2 Proof of (C.3)

To prove (C.3), first note that the list $\mathcal{B}'$ of $\mathbf{WM}'$ is defined by the following propositions:

a. $[\!|_{s_\perp} 1 |\!]^{s_\perp} \in (\mathcal{B}')^1$ and $[\!|_{s_\not\perp} 1 |\!]^{s_\not\perp} \in (\mathcal{B}')^1$ for all $s \in S$;

b. $[\!|_{s_\perp} a |\!]^{s'_\perp} \in (\mathcal{B}')^1$ for all $s, s' \in S$ and $a \in \Sigma_{\mathbf{i}}$ such that there exists a $s'' \in S$ such that $s'' \in \varepsilon\mathfrak{t}^*_{\mathcal{A}}(s')$ and $[\!|_s a |\!]^{s''} \in \mathcal{B}^1$;

c. $[\!|_{s_\not\perp} a |\!]^{s'_\not\perp} \in (\mathcal{B}')^1$ for all $s, s' \in S$ and $a \in \Sigma_{\mathbf{i}}$ such that $[\!|_s a |\!]^{s'} \in \mathcal{B}^1$;

d. $[\!|_{s_\perp} c{\downarrow_t}_{\not\perp} {\uparrow^{t'}}_{\not\perp} r |\!]^{s'_\perp} \in (\mathcal{B}')^2$ for all $s, s', t, t' \in S$, $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$ such that there exists $s'' \in S$ such that $s'' \in \varepsilon\mathfrak{t}^*_{\mathcal{A}}(s')$ and $[\!|_s c{\downarrow_t} {\uparrow^{t'}} r |\!]^{s''} \in \mathcal{B}^2$;

e. $[\!|_{s_\not\perp} c{\downarrow_t}_{\not\perp} {\uparrow^{t'}}_{\not\perp} r |\!]^{s'_\not\perp} \in (\mathcal{B}')^2$ for all $s, s', t, t' \in S$, $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$ such that $[\!|_s c{\downarrow_t} {\uparrow^{t'}} r |\!]^{s'} \in \mathcal{B}^2$.

It is also useful to note that for all $s, s' \in S$,

$$\vdash \underset{s}{(\!|} \mathcal{B} \overset{s'}{|\!)} = \underset{s_{\mathfrak{c}}}{(\!|} \mathcal{C} \overset{s'_{\mathfrak{c}}}{|\!)} \quad . \tag{C.15}$$

For the case $\leqslant$ of (C.15), we use (3.37) with the substitution function $h_1 : S \to S \cup \{s_{\mathfrak{c}} \mid s \in S\}$ defined by $h_1(s) := s_{\mathfrak{c}}$, and we note that $\hat{h_1}(\mathcal{B})$ shrinked with idempotency of blocks and swapping of blocks gives a sublist of $\mathcal{C}$. So, the case $\leqslant$ of (C.15) follows by (3.26) and (3.30).

For the case $\geqslant$ of (C.15), we use (3.37) with the substitution function $h_2 : S \cup \{s_{\mathfrak{c}} \mid s \in S\} \to S$ defined by $h_2(s) := s$ and $h_2(s_{\mathfrak{c}}) := s$, and we note that $\hat{h_2}(\mathcal{C})$ shrinked with idempotency of blocks and swapping of blocks gives exactly $\mathcal{B}$.

**Proof of the case $\leqslant$ of (C.3):** We use axiom (3.14) with

$$\begin{aligned} s_{(s_\perp, s'_\perp)} &:= (\mathbf{WM}_{\mathfrak{c}} \bullet \varepsilon^\star_\perp)^\star[s, s'] \ , \\ s_{(s_\perp, s'_\not\perp)} &:= 0 \ , \\ s_{(s_\not\perp, s'_\perp)} &:= 0 \ , \\ s_{(s_\not\perp, s'_\not\perp)} &:= \underset{s}{(\!|} \mathcal{B} \overset{s'}{|\!)} \ , \end{aligned}$$

for all $s, s' \in S$. So, using also the definition of $\mathcal{B}'$, it suffices to prove, for all $s, s' \in S$,

$$(\wedge\, m \mid \underset{s_\perp}{[} m \overset{s'_\perp}{]} \in (\mathcal{B}')^1 : m \leqslant (\mathbf{WM}_{\mathfrak{c}} \bullet \varepsilon^\star_\perp)^\star[s, s']) \ , \tag{C.16}$$

$$(\wedge\, m \mid \underset{s_\not\perp}{[} m \overset{s'_\not\perp}{]} \in (\mathcal{B}')^1 : m \leqslant \underset{s}{(\!|} \mathcal{B} \overset{s'}{|\!)}) \ , \tag{C.17}$$

$$(\wedge\, m, v \mid \underset{s_\perp}{[} m \overset{v_\perp}{]} \in (\mathcal{B}')^1 : m \cdot (\mathbf{WM}_{\mathfrak{c}} \bullet \varepsilon^\star_\perp)^\star[v, s'] \leqslant (\mathbf{WM}_{\mathfrak{c}} \bullet \varepsilon^\star_\perp)^\star[s, s']) \ , \tag{C.18}$$

$$(\wedge\, m, v \mid \underset{s_\perp}{[} m \overset{v_\perp}{]} \in (\mathcal{B}')^1 : m \cdot 0 \leqslant 0) \ , \tag{C.19}$$

$$(\wedge\, m, v \mid \underset{s_\not\perp}{[} m \overset{v_\not\perp}{]} \in (\mathcal{B}')^1 : m \cdot 0 \leqslant 0) \ , \tag{C.20}$$

$$(\wedge\, m, v \mid \underset{s_\not\perp}{[} m \overset{v_\not\perp}{]} \in (\mathcal{B}')^1 : m \cdot \underset{v}{(\!|} \mathcal{B} \overset{s'}{|\!)} \leqslant \underset{s}{(\!|} \mathcal{B} \overset{s'}{|\!)}) \ , \tag{C.21}$$

$$(\wedge\, c, z, r, w \mid \underset{s_\perp}{[} c \underset{z_\not\perp}{\downarrow} \overset{w_\not\perp}{\uparrow} r \overset{s'_\perp}{]} \in (\mathcal{B}')^2 : c \cdot \underset{z}{(\!|} \mathcal{B} \overset{w}{|\!)} \cdot r \leqslant (\mathbf{WM}_{\mathfrak{c}} \bullet \varepsilon^\star_\perp)^\star[s, s']) \ , \tag{C.22}$$

$$(\wedge\, c, z, r, w \mid \underset{s_\not\perp}{[} c \underset{z_\not\perp}{\downarrow} \overset{w_\not\perp}{\uparrow} r \overset{s'_\not\perp}{]} \in (\mathcal{B}')^2 : c \cdot \underset{z}{(\!|} \mathcal{B} \overset{w}{|\!)} \cdot r \leqslant \underset{s}{(\!|} \mathcal{B} \overset{s'}{|\!)}) \ , \tag{C.23}$$

$$(\wedge\, c,z,r,w,v \mid \; [\!]_{s_\perp}\; c \downarrow_{z_\not} \overset{w_\not}{\uparrow} r \;[\!]^{v_\perp} \in (\mathcal{B}')^2 : c \cdot (\!|\mathcal{B}|\!)^{w}_{z} \cdot r \cdot (\mathbf{WM}_{\mathfrak{c}} \bullet \varepsilon^\star_\perp)^\star[v,s'] \leqslant (\mathbf{WM}_{\mathfrak{c}} \bullet \varepsilon^\star_\perp)^\star[s,s'])\ , \tag{C.24}$$

$$(\wedge\, c,z,r,w,v \mid \; [\!]_{s_\perp}\; c \downarrow_{z_\not} \overset{w_\not}{\uparrow} r \;[\!]^{v_\perp} \in (\mathcal{B}')^2 : c \cdot (\!|\mathcal{B}|\!)^{w}_{z} \cdot r \cdot 0 \leqslant 0)\ , \tag{C.25}$$

$$(\wedge\, c,z,r,w,v \mid \; [\!]_{s_\not}\; c \downarrow_{z_\not} \overset{w_\not}{\uparrow} r \;[\!]^{v_\not} \in (\mathcal{B}')^2 : c \cdot (\!|\mathcal{B}|\!)^{w}_{z} \cdot r \cdot 0 \leqslant 0)\ , \tag{C.26}$$

$$(\wedge\, c,z,r,w,v \mid \; [\!]_{s_\not}\; c \downarrow_{z_\not} \overset{w_\not}{\uparrow} r \;[\!]^{v_\not} \in (\mathcal{B}')^2 : c \cdot (\!|\mathcal{B}|\!)^{w}_{z} \cdot r \cdot (\!|\mathcal{B}|\!)^{s'}_{v} \leqslant (\!|\mathcal{B}|\!)^{s'}_{s})\ . \tag{C.27}$$

For inequations (C.16), we suppose $[\!]_{s_\perp}\; m \;[\!]^{s'_\perp} \in (\mathcal{B}')^1$ and we prove

$$m \leqslant (\mathbf{WM}_{\mathfrak{c}} \bullet \varepsilon^\star_\perp)^\star[s,s']\ .$$

Let us prove this by case analysis on $m$. The case $m = 0$ does not exist by the definition of $\mathcal{B}'$.

For the case $m = 1$, by condition a of the definition of $\mathcal{B}'$, $s_\perp = s'_\perp$. So, it suffices to prove that

$$1 \leqslant (\mathbf{WM}_{\mathfrak{c}} \bullet \varepsilon^\star_\perp)^\star[s,s]\ .$$

This is direct by Kleene algebra stating that $\mathbf{I} \leqslant \mathbf{A}^\star$ for any square matrix $\mathbf{A}$ and by the definition of $\mathbf{I}$.

For the case $m \in \Sigma_{\mathbf{i}}$, by Kleene algebra, it suffices to prove

$$m \leqslant (\mathbf{WM}_{\mathfrak{c}} \bullet \varepsilon^\star_\perp)[s,s']\ .$$

By the definition of $\bullet$, it suffices to prove

$$m \leqslant (\sum t \mid t \in S : \mathbf{WM}_{\mathfrak{c}}[s,t] \cdot \varepsilon^\star_\perp[t,s'])\ .$$

By the definition of $\mathbf{WM}_{\mathfrak{c}}$, it suffices to prove

$$m \leqslant (\sum t \mid t \in S : (\!|\mathcal{C}|\!)^{t}_{s} \cdot \varepsilon^\star_\perp[t,s'])\ .$$

By the definition of $\varepsilon^\star_\perp$, each entry of this matrix is either 1 or 0 (modulo the axioms of Kleene algebra). So, by the definition of $\varepsilon\mathsf{t}^*_{\mathcal{A}}(s')$ and Kleene algebra, it suffices to prove

$$m \leqslant (\sum t \mid t \in \varepsilon\mathsf{t}^*_{\mathcal{A}}(s') : (\!|\mathcal{C}|\!)^{t}_{s})\ .$$

By condition b of the definition of $\mathcal{B}'$, there exists a $s'' \in S$ such that $s'' \in \varepsilon \mathsf{t}^*_{\mathcal{A}}(s')$ and $\lfloor_s m \rceil^{s''} \in \mathcal{B}^1$. So, $\lfloor_s m \rceil^{s''} \in \mathcal{C}^1$ by condition b of the definition of $\mathcal{C}$. Then, the proof follows from axiom (3.2) and Kleene algebra.

For inequations (C.17), we suppose $\lfloor_{s_{\not\perp}} m \rceil^{s'_{\not\perp}} \in (\mathcal{B}')^1$ and we prove

$$m \leqslant (\!|\, \mathcal{B} \,|\!)^{s'}_{s} \quad .$$

Let us prove this by case analysis on $m$. The case $m = 0$ does not exist by the definition of $\mathcal{B}'$. For the case $m = 1$, by condition a of the definition of $\mathcal{B}'$, $s_{\not\perp} = s'_{\not\perp}$, and by the definition of $\mathcal{B}$, $\lfloor_s 1 \rceil^s \in \mathcal{B}^1$. So, the proof follows from axiom (3.2). For the case $m \in \Sigma_{\mathbf{i}}$, by condition c of the definition of $\mathcal{B}'$, $\lfloor_s m \rceil^{s'} \in \mathcal{B}^1$. Then, the proof follows from axiom (3.2).

For inequations (C.18), we suppose $\lfloor_{s_\perp} m \rceil^{v_\perp} \in (\mathcal{B}')^1$ and we prove

$$m \cdot (\mathbf{WM}_{\mathbf{c}} \bullet \varepsilon^\star_\perp)^\star[v, s'] \leqslant (\mathbf{WM}_{\mathbf{c}} \bullet \varepsilon^\star_\perp)^\star[s, s'] \quad . \tag{C.28}$$

We first prove that

$$(\mathbf{WM}_{\mathbf{c}} \bullet \varepsilon^\star_\perp)^\star[s, v] \cdot (\mathbf{WM}_{\mathbf{c}} \bullet \varepsilon^\star_\perp)^\star[v, s'] \leqslant (\mathbf{WM}_{\mathbf{c}} \bullet \varepsilon^\star_\perp)^\star[s, s'] \quad . \tag{C.29}$$

We are able to prove it.

$$\begin{array}{ll} & (\mathbf{WM}_{\mathbf{c}} \bullet \varepsilon^\star_\perp)^\star[s, s'] \\ = & \qquad \{\!\{ \text{ Kleene algebra } (p^* = p^* p^*) \,\}\!\} \\ & ((\mathbf{WM}_{\mathbf{c}} \bullet \varepsilon^\star_\perp)^\star \bullet (\mathbf{WM}_{\mathbf{c}} \bullet \varepsilon^\star_\perp)^\star)[s, s'] \\ = & \qquad \{\!\{ \text{ Definition of } \bullet \,\}\!\} \\ & (\sum t \mid t \in S : (\mathbf{WM}_{\mathbf{c}} \bullet \varepsilon^\star_\perp)^\star[s, t] \cdot (\mathbf{WM}_{\mathbf{c}} \bullet \varepsilon^\star_\perp)^\star[t, s']) \\ \geqslant & \qquad \{\!\{ \text{ Hypothesis: } v \in S \text{ \& Kleene algebra } \,\}\!\} \\ & (\mathbf{WM}_{\mathbf{c}} \bullet \varepsilon^\star_\perp)^\star[s, v] \cdot (\mathbf{WM}_{\mathbf{c}} \bullet \varepsilon^\star_\perp)^\star[v, s'] \end{array}$$

We now prove (C.28). By (C.29) and Kleene algebra (mainly, monotonicity of $\cdot$), it suffices to prove

$$m \leqslant (\mathbf{WM}_{\mathbf{c}} \bullet \varepsilon^\star_\perp)^\star[s, v] \quad .$$

The proof follows from inequations (C.16).

Inequations (C.19) and (C.20) are direct from zero of $\cdot$ and reflexivity of $\leqslant$.

For inequations (C.21), we suppose $\lfloor_{s_{\not\perp}}\, m \,\rceil^{v_{\not\perp}} \in (\mathcal{B}')^1$ and we prove

$$m \cdot \underset{v}{(\!|}\, \mathcal{B} \,\overset{s'}{|\!)} \leqslant \underset{s}{(\!|}\, \mathcal{B} \,\overset{s'}{|\!)} \quad .$$

Let us prove this by case analysis on $m$. The case $m = 0$ does not exist by the definition of $\mathcal{B}'$. For the case $m = 1$, by condition a of the definition of $\mathcal{B}'$, $s_{\not\perp} = v_{\not\perp}$. So, it suffices to prove

$$1 \cdot \underset{s}{(\!|}\, \mathcal{B} \,\overset{s'}{|\!)} \leqslant \underset{s}{(\!|}\, \mathcal{B} \,\overset{s'}{|\!)} \quad .$$

This is direct by Kleene algebra. For the case $m \in \Sigma_{\mathbf{i}}$, by condition c of the definition of $\mathcal{B}'$, $\lfloor_s\, m \,\rceil^v \in \mathcal{B}^1$. By axiom (3.2) and Kleene algebra, it suffices to prove

$$\underset{s}{(\!|}\, \mathcal{B} \,\overset{v}{|\!)} \cdot \underset{v}{(\!|}\, \mathcal{B} \,\overset{s'}{|\!)} \leqslant \underset{s}{(\!|}\, \mathcal{B} \,\overset{s'}{|\!)} \quad .$$

This is direct by axiom (3.4).

For inequations (C.22), we suppose $\lfloor_{s_\perp}\, c{\downarrow_{z_{\not\perp}}} {\uparrow^{w_{\not\perp}}}\, r \,\rceil^{s'_\perp} \in (\mathcal{B}')^2$ and we prove

$$c \cdot \underset{z}{(\!|}\, \mathcal{B} \,\overset{w}{|\!)} \cdot r \leqslant (\mathbf{WM}_{\mathfrak{c}} \bullet \varepsilon^\star_\perp)^\star[s, s'] \quad .$$

By Kleene algebra, it suffices to prove

$$c \cdot \underset{z}{(\!|}\, \mathcal{B} \,\overset{w}{|\!)} \cdot r \leqslant (\mathbf{WM}_{\mathfrak{c}} \bullet \varepsilon^\star_\perp)[s, s'] \quad .$$

By the definition of $\bullet$, it suffices to prove

$$c \cdot \underset{z}{(\!|}\, \mathcal{B} \,\overset{w}{|\!)} \cdot r \leqslant (\textstyle\sum\, t \mid t \in S : \mathbf{WM}_{\mathfrak{c}}[s, t] \cdot \varepsilon^\star_\perp[t, s']) \quad .$$

By the definition of $\mathbf{WM}_{\mathfrak{c}}$ and (C.15), it suffices to prove

$$c \cdot \underset{z_{\mathfrak{c}}}{(\!|}\, \mathcal{C} \,\overset{w_{\mathfrak{c}}}{|\!)} \cdot r \leqslant (\textstyle\sum\, t \mid t \in S : \underset{s}{(\!|}\, \mathcal{C} \,\overset{t}{|\!)} \cdot \varepsilon^\star_\perp[t, s']) \quad .$$

By the definition of $\varepsilon^\star_\perp$, each entry of this matrix is either 1 or 0 (modulo the axioms of Kleene algebra). So, by the definition of $\varepsilon\mathfrak{t}^*_{\mathcal{A}}(s')$ and Kleene algebra, it suffices to prove

$$c \cdot \underset{z_{\mathfrak{c}}}{(\!|}\, \mathcal{C} \,\overset{w_{\mathfrak{c}}}{|\!)} \cdot r \leqslant (\textstyle\sum\, t \mid t \in \varepsilon\mathfrak{t}^*_{\mathcal{A}}(s') : \underset{s}{(\!|}\, \mathcal{C} \,\overset{t}{|\!)}) \quad .$$

Since $\lfloor_{s_\perp}\, c{\downarrow_{z_{\not\perp}}} {\uparrow^{w_{\not\perp}}}\, r \,\rceil^{s'_\perp} \in (\mathcal{B}')^2$, by condition d of the definition of $\mathcal{B}'$, there exists $s'' \in S$ such that $s'' \in \varepsilon\mathfrak{t}^*_{\mathcal{A}}(s')$ and $\lfloor_s\, c{\downarrow_z} {\uparrow^w}\, r \,\rceil^{s''} \in \mathcal{B}^2$. So, $\lfloor_s\, c{\downarrow_{z_{\mathfrak{c}}}} {\uparrow^{w_{\mathfrak{c}}}}\, r \,\rceil^{s''} \in \mathcal{C}^2$ by condition c of the definition of $\mathcal{C}$. Then, the proof follows from axiom (3.3) and Kleene algebra.

For inequations (C.23), we suppose $[_{s_\not} \, c{\downarrow_{z_\not}} \uparrow^{w_\not} r \,]^{s'_\not} \in (\mathcal{B}')^2$ and we prove

$$c \cdot \underset{z}{(\!|} \mathcal{B} \overset{w}{|\!)} \cdot r \leqslant \underset{s}{(\!|} \mathcal{B} \overset{s'}{|\!)} \quad .$$

By condition e of the definition of $\mathcal{B}'$, $[_s \, c{\downarrow_z} \uparrow^w r \,]^{s'} \in \mathcal{B}^2$. So, the proof follows from axiom (3.3).

For inequations (C.24), we suppose $[_{s_\bot} \, c{\downarrow_{z_\not}} \uparrow^{w_\not} r \,]^{v_\bot} \in (\mathcal{B}')^2$ and we prove

$$c \cdot \underset{z}{(\!|} \mathcal{B} \overset{w}{|\!)} \cdot r \cdot (\mathbf{WM_c} \bullet \varepsilon^\star_\bot)^\star[v, s'] \leqslant (\mathbf{WM_c} \bullet \varepsilon^\star_\bot)^\star[s, s'] \quad .$$

By (C.29) and Kleene algebra (mainly, monotonicity of $\cdot$), it suffices to prove

$$c \cdot \underset{z}{(\!|} \mathcal{B} \overset{w}{|\!)} \cdot r \leqslant (\mathbf{WM_c} \bullet \varepsilon^\star_\bot)^\star[s, v] \quad .$$

The proof follows from inequations (C.22).

Inequations (C.25) and (C.26) are direct from zero of $\cdot$ and reflexivity of $\leqslant$.

For inequations (C.27), we suppose $[_{s_\not} \, c{\downarrow_{z_\not}} \uparrow^{w_\not} r \,]^{v_\not} \in (\mathcal{B}')^2$ and we prove

$$c \cdot \underset{z}{(\!|} \mathcal{B} \overset{w}{|\!)} \cdot r \cdot \underset{v}{(\!|} \mathcal{B} \overset{s'}{|\!)} \leqslant \underset{s}{(\!|} \mathcal{B} \overset{s'}{|\!)} \quad .$$

By condition e of the definition of $\mathcal{B}'$, $[_s \, c{\downarrow_z} \uparrow^w r \,]^v \in \mathcal{B}^2$. So, by axiom (3.3) and Kleene algebra, it suffices to prove

$$\underset{s}{(\!|} \mathcal{B} \overset{v}{|\!)} \cdot \underset{v}{(\!|} \mathcal{B} \overset{s'}{|\!)} \leqslant \underset{s}{(\!|} \mathcal{B} \overset{s'}{|\!)} \quad .$$

This is direct by axiom (3.4).

**Proof of the case $\geqslant$ of (C.3):** First note that, for all $s, s' \in S$,

$$\vdash 0 \leqslant \mathbf{WM}'[s_\bot, s'_\not] \qquad \text{and} \qquad \vdash 0 \leqslant \mathbf{WM}'[s_\not, s'_\bot]$$

by the fact that 0 is the minimum of the algebra. Also, for all $s, s' \in S$,

$$\vdash \underset{s}{(\!|} \mathcal{B} \overset{s'}{|\!)} \leqslant \underset{s_\not}{(\!|} \mathcal{B}' \overset{s'_\not}{|\!)}$$

since the function $g : S \to \{t_\perp, t_\not\perp \mid t \in S\}$ defined by $g(t') := t'_\not\perp$ for all $t' \in S$ is such that $\hat{g}(\mathcal{B})$ gives a sublist of $\mathcal{B}'$ (by definition of $\mathcal{B}'$, conditions a, c and e), and so

$$\vdash \mathop{(\!|}_{s} \mathcal{B} \mathop{|\!)}^{s'} \leqslant \mathop{(\!|}_{s_\not\perp} \hat{g}(\mathcal{B}) \mathop{|\!)}^{s'_\not\perp}$$

by (3.37), and

$$\vdash \mathop{(\!|}_{s_\not\perp} \hat{g}(\mathcal{B}) \mathop{|\!)}^{s'_\not\perp} \leqslant \mathop{(\!|}_{s_\not\perp} \mathcal{B}' \mathop{|\!)}^{s'_\not\perp}$$

by (3.26) and (3.30).

Let $\mathbf{WM}''$ be the quadrant matrix of $\mathbf{WM}'$ for the entries $\{t_\perp \mid t \in S\} \times \{t_\perp \mid t \in S\}$. In other words, for all $t, t' \in S$,

$$\mathbf{WM}''[t_\perp, t'_\perp] := \mathbf{WM}'[t_\perp, t'_\perp] \ .$$

To finish the proof of the case $\geqslant$ of (C.3), it remains to prove that

$$\vdash (\mathbf{WM}_\mathfrak{c} \bullet \varepsilon^\star_\perp)^\star \leqslant \mathbf{WM}'' \ .$$

By $^*$-induction and Kleene algebra, it suffices to show independently that

$$\vdash \mathbf{I} \leqslant \mathbf{WM}'' \qquad \text{and} \qquad \vdash \mathbf{WM}_\mathfrak{c} \bullet \varepsilon^\star_\perp \bullet \mathbf{WM}'' \leqslant \mathbf{WM}'' \ .$$

The first inequation is trivial by Lemma 4.5, property i. For the second inequation, note that the proof is direct if it can be proved that

$$\vdash \mathbf{WM}_\mathfrak{c} \bullet \varepsilon^\star_\perp \leqslant \mathbf{WM}'' \ . \tag{C.30}$$

An example of such a proof is

$$\begin{array}{ll} & \mathbf{WM}_\mathfrak{c} \bullet \varepsilon^\star_\perp \bullet \mathbf{WM}'' \\ \leqslant & \qquad \{\!\{ \text{ Inequation (C.30) \& Monotonicity of } \cdot \ \}\!\} \\ & \mathbf{WM}'' \bullet \mathbf{WM}'' \\ = & \qquad \{\!\{ \text{ Lemma 4.5, property ii } \}\!\} \\ & \mathbf{WM}'' \ . \end{array}$$

It remains to prove (C.30). So, for all $s, s' \in S$, we prove that

$$\vdash (\mathbf{WM}_\mathfrak{c} \bullet \varepsilon^\star_\perp)[s, s'] \leqslant \mathbf{WM}''[s_\perp, s'_\perp] \ .$$

By definition of $\bullet$, $\varepsilon\mathsf{t}^*_{\mathcal{A}}$, $\mathbf{WM}_{\mathfrak{c}}$ and $\mathbf{WM}''$, and Kleene algebra, it suffices to prove independently that

$$\vdash \underset{s}{(\!|}\, \mathcal{C} \,\underset{}{|\!)}^{t} \leqslant \underset{s_\perp}{(\!|}\, \mathcal{B}' \,\underset{}{|\!)}^{s'_\perp}$$

for each $t \in \varepsilon\mathsf{t}^*_{\mathcal{A}}(s')$. By (3.37) with $g' : S \cup \{u_{\mathfrak{c}} \mid u \in S\} \to \{u_\perp, u_{\not\perp} \mid u \in S\}$ defined by $g'(u) := u_\perp$ and $g'(u_{\mathfrak{c}}) := u_{\not\perp}$ for all $u \in S$,

$$\vdash \underset{s}{(\!|}\, \mathcal{C} \,\underset{}{|\!)}^{t} \leqslant \underset{s_\perp}{(\!|}\, \widehat{g'}(\mathcal{C}) \,\underset{}{|\!)}^{t_\perp} \quad .$$

Furthermore,

$$\begin{aligned}
& (\!|_{s_\perp}\, \widehat{g'}(\mathcal{C}) \,|\!)^{t_\perp} \\
= \quad & \{\!\{ \text{ Equation (3.25) \& Definition of } \mathcal{C},\ g' \text{ and } \widehat{g'} \ \}\!\} \\
& (\sum m \mid m \in \Sigma_{\mathbf{i}} \wedge [_{s_\perp}\, m\, ]^{t_\perp} \in (\widehat{g'}(\mathcal{C}))^1 : m) \\
& +(\sum m, v \mid m \in \Sigma_{\mathbf{i}} \wedge [_{v_\perp}\, m\, ]^{t_\perp} \in (\widehat{g'}(\mathcal{C}))^1 : (\!|_{s_\perp}\, \widehat{g'}(\mathcal{C}) \,|\!)^{v_\perp} \cdot m) \\
& +(\sum c, z, r, w \mid [_{s_\perp}\, c{\downarrow_{z_{\not\perp}}} {\uparrow^{w_{\not\perp}}}\, r\, ]^{t_\perp} \in (\widehat{g'}(\mathcal{C}))^2 : c \cdot (\!|_{z_{\not\perp}}\, \widehat{g'}(\mathcal{C}) \,|\!)^{w_{\not\perp}} \cdot r) \\
& +(\sum c, z, r, w, v \mid [_{v_\perp}\, c{\downarrow_{z_{\not\perp}}} {\uparrow^{w_{\not\perp}}}\, r\, ]^{t_\perp} \in (\widehat{g'}(\mathcal{C}))^2 : (\!|_{s_\perp}\, \widehat{g'}(\mathcal{C}) \,|\!)^{v_\perp} \cdot c \cdot (\!|_{z_{\not\perp}}\, \widehat{g'}(\mathcal{C}) \,|\!)^{w_{\not\perp}} \cdot r) \\
\leqslant \quad & \{\!\{ \text{ By definition of } \mathcal{C},\ \mathcal{B}' \text{ and } \widehat{g'},\ \text{the list } \widehat{g'}(\mathcal{C}) \text{ is a sublist of } \mathcal{B}'. \\
& \quad \text{\& Inequations (3.26) and (3.30) \& Monotonicity of } \cdot \text{ and } + \ \}\!\} \\
& (\sum m \mid m \in \Sigma_{\mathbf{i}} \wedge [_{s_\perp}\, m\, ]^{t_\perp} \in (\mathcal{B}')^1 : m) \\
& +\ (\sum m, v \mid m \in \Sigma_{\mathbf{i}} \wedge [_{v_\perp}\, m\, ]^{t_\perp} \in (\mathcal{B}')^1 : (\!|_{s_\perp}\, \mathcal{B}' \,|\!)^{v_\perp} \cdot m) \\
& +\ (\sum c, z, r, w \mid [_{s_\perp}\, c{\downarrow_{z_{\not\perp}}} {\uparrow^{w_{\not\perp}}}\, r\, ]^{t_\perp} \in (\mathcal{B}')^2 : c \cdot (\!|_{z_{\not\perp}}\, \mathcal{B}' \,|\!)^{w_{\not\perp}} \cdot r) \\
& +\ (\sum c, z, r, w, v \mid [_{v_\perp}\, c{\downarrow_{z_{\not\perp}}} {\uparrow^{w_{\not\perp}}}\, r\, ]^{t_\perp} \in (\mathcal{B}')^2 : (\!|_{s_\perp}\, \mathcal{B}' \,|\!)^{v_\perp} \cdot c \cdot (\!|_{z_{\not\perp}}\, \mathcal{B}' \,|\!)^{w_{\not\perp}} \cdot r) \\
\leqslant \quad & \{\!\{ \text{ Hypothesis: } t \in \varepsilon\mathsf{t}^*_{\mathcal{A}}(s') \text{ \& Definition of } \mathcal{B}' \text{: if } m \in \Sigma_{\mathbf{i}},\ [_{v_\perp}\, m\, ]^{t_\perp} \in (\mathcal{B}')^1 \\
& \quad \text{and } t \in \varepsilon\mathsf{t}^*_{\mathcal{A}}(s'),\ \text{then } [_{v_\perp}\, m\, ]^{s'_\perp} \in (\mathcal{B}')^1.\ \text{Also, if } [_{v_\perp}\, c{\downarrow_{z_{\not\perp}}} {\uparrow^{w_{\not\perp}}}\, r\, ]^{t_\perp} \in (\mathcal{B}')^2 \\
& \quad \text{and } t \in \varepsilon\mathsf{t}^*_{\mathcal{A}}(s'),\ \text{then } [_{v_\perp}\, c{\downarrow_{z_{\not\perp}}} {\uparrow^{w_{\not\perp}}}\, r\, ]^{s'_\perp} \in (\mathcal{B}')^2. \\
& \quad \text{\& Quantification laws } \}\!\} \\
& (\sum m \mid [_{s_\perp}\, m\, ]^{s'_\perp} \in (\mathcal{B}')^1 : m) \\
& +\ (\sum m, v \mid [_{v_\perp}\, m\, ]^{s'_\perp} \in (\mathcal{B}')^1 : (\!|_{s_\perp}\, \mathcal{B}' \,|\!)^{v_\perp} \cdot m) \\
& +\ (\sum c, z, r, w \mid [_{s_\perp}\, c{\downarrow_{z_{\not\perp}}} {\uparrow^{w_{\not\perp}}}\, r\, ]^{s'_\perp} \in (\mathcal{B}')^2 : c \cdot (\!|_{z_{\not\perp}}\, \mathcal{B}' \,|\!)^{w_{\not\perp}} \cdot r) \\
& +\ (\sum c, z, r, w, v \mid [_{v_\perp}\, c{\downarrow_{z_{\not\perp}}} {\uparrow^{w_{\not\perp}}}\, r\, ]^{s'_\perp} \in (\mathcal{B}')^2 : (\!|_{s_\perp}\, \mathcal{B}' \,|\!)^{v_\perp} \cdot c \cdot (\!|_{z_{\not\perp}}\, \mathcal{B}' \,|\!)^{w_{\not\perp}} \cdot r) \\
= \quad & \{\!\{ \text{ Equation (3.25) } \}\!\} \\
& (\!|_{s_\perp}\, \mathcal{B}' \,|\!)^{s'_\perp} \ .
\end{aligned}$$

## C.2 Step 2: Elimination of $\varepsilon$-Transitions of the Form $(s, \varepsilon, d; s', \bot)$ for All $s, s' \in S$ and $d \in \Gamma$

**Theorem C.2** (Elimination of $\varepsilon$-transitions of the form $(s, \varepsilon, d; s', \bot)$)**.** *Let $\Sigma_{\mathbf{i}}$, $\Sigma_{\mathbf{c}}$ and $\Sigma_{\mathbf{r}}$ be three disjoint finite sets such that at least one of them is nonempty. Let $\mathcal{A} := (S, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma \cup \{\bot\}, \delta, I, F)$ be a semi-visibly pushdown automaton such that $\delta$ does not contain $\varepsilon$-transitions of the form $(s, \varepsilon, \bot; s', \bot)$ for all $s, s' \in S$. Let*

$$(\overrightarrow{I}, \mathbf{WM}, \mathbf{T}_{\mathsf{c}}, \mathbf{T}_{\bot}, \varepsilon_{\bot}, \varepsilon_{\not\bot}, \overrightarrow{F})$$

*be the algebraic encoding of $\mathcal{A}$. Then, there exists a semi-visibly pushdown automaton $\mathcal{A}' := (S', \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma' \cup \{\bot\}, \delta', I', F')$ having an algebraic encoding*

$$(\overrightarrow{I'}, \mathbf{WM}', \mathbf{T}'_{\mathsf{c}}, \mathbf{T}'_{\bot}, \varepsilon'_{\bot}, \varepsilon'_{\not\bot}, \overrightarrow{F'})$$

*such that $\delta'$ does not contain $\varepsilon$-transitions of the form $(s, \varepsilon, d; s', \bot)$ for all $s, s' \in S'$ and $d \in \Gamma' \cup \{\bot\}$ and*

$$\begin{aligned} &\vdash \overrightarrow{I}^{t} \bullet (\mathbf{T}_{\bot} + \mathbf{WM} + \varepsilon_{\bot} + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{+} \bullet \varepsilon_{\not\bot})^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet \overrightarrow{F} \\ &\quad = \overrightarrow{I'}^{t} \bullet (\mathbf{T}'_{\bot} + \mathbf{WM}' + \varepsilon'_{\bot} + (\mathbf{T}'_{\mathsf{c}} \bullet \mathbf{WM}')^{+} \bullet \varepsilon'_{\not\bot})^{\star} \bullet (\mathbf{T}'_{\mathsf{c}} \bullet \mathbf{WM}')^{\star} \bullet \overrightarrow{F'} \ . \end{aligned} \tag{C.31}$$

*So, $\mathcal{A}'$ is a visibly pushdown automaton and $\mathcal{A}'$ has at most thrice the number of states of $\mathcal{A}$.*

*Proof.* The automaton has states of the form $s_{\mathfrak{a}_{\bot}}$, $s_{\mathfrak{a}_{\not\bot}}$ and $s_{\mathfrak{p}_{\not\bot}}$ for each $s \in S$. The idea is just to encode in a state an information stating if the stack is "allowed" to be read normally and is empty (this is represented by $s_{\mathfrak{a}_{\bot}}$) or if the stack is "allowed" to be read normally and is nonempty (this is represented by $s_{\mathfrak{a}_{\not\bot}}$) or if it must be "protected" when reading symbols from the nonempty stack (this is represented by $s_{\mathfrak{p}_{\not\bot}}$). A state $s_{\mathfrak{p}_{\not\bot}}$ protecting the reading of the stack is just a way to say that the values of the symbols on the stack are irrelevant. A value of a symbol on the stack becomes irrelevant when simulating a transition of the form $(t, \varepsilon, d; t', \bot)$ which protects the stack from being read. The idea is that any transition of the form $(t, r, \bot; t', \bot)$ is simulated in $t_{\mathfrak{p}_{\not\bot}}$ by saturation of the transitions over the possible stack symbols.

Define the visibly pushdown automaton

$$\begin{aligned} \mathcal{A}' := (&\{s_{\mathfrak{a}_{\bot}}, s_{\mathfrak{a}_{\not\bot}}, s_{\mathfrak{p}_{\not\bot}} \mid s \in S\}, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \{\mathfrak{a}_{\bot}, \mathfrak{a}_{\not\bot}, \mathfrak{p}_{\not\bot}\} \times \Gamma \cup \{\bot\}, \delta', \{i_{\mathfrak{a}_{\bot}} \mid i \in I\}, \\ &\{f_{\mathfrak{a}_{\bot}}, f_{\mathfrak{a}_{\not\bot}}, f_{\mathfrak{p}_{\not\bot}} \mid f \in F\}) \end{aligned}$$

where

$$\begin{aligned}
\delta' = \{&(s_{\mathfrak{a}_\perp}, a, \lambda; s'_{\mathfrak{a}_\perp}, \lambda), (s_{\mathfrak{a}_\not\perp}, a, \lambda; s'_{\mathfrak{a}_\not\perp}, \lambda), (s_{\mathfrak{p}_\not\perp}, a, \lambda; s'_{\mathfrak{p}_\not\perp}, \lambda) \mid a \in \Sigma_{\mathbf{i}} \\
&\quad \wedge\ (s, a, \lambda; s', \lambda) \in \delta\} \\
\cup\ &\{(s_{\mathfrak{a}_\not\perp}, a, \lambda; s'_{\mathfrak{p}_\not\perp}, \lambda) \mid a \in \Sigma_{\mathbf{i}} \\
&\quad \wedge\ (\exists\ s'' \mid (\forall\ d \mid d \in \Gamma : (s'', \varepsilon, d; s', \perp) \in \delta) : (s, a, \lambda; s'', \lambda) \in \delta)\} \\
\cup\ &\{(s_{\mathfrak{a}_\perp}, c, \lambda; s'_{\mathfrak{a}_\not\perp}, (\mathfrak{a}_\perp, d)), (s_{\mathfrak{a}_\not\perp}, c, \lambda; s'_{\mathfrak{a}_\not\perp}, (\mathfrak{a}_\not\perp, d)), (s_{\mathfrak{p}_\not\perp}, c, \lambda; s'_{\mathfrak{a}_\not\perp}, (\mathfrak{p}_\not\perp, d)) \mid c \in \Sigma_{\mathbf{c}} \\
&\quad \wedge\ (s, c, \lambda; s', d) \in \delta\} \\
\cup\ &\{(s_{\mathfrak{a}_\perp}, c, \lambda; s'_{\mathfrak{p}_\not\perp}, (\mathfrak{a}_\perp, d)), (s_{\mathfrak{a}_\not\perp}, c, \lambda; s'_{\mathfrak{p}_\not\perp}, (\mathfrak{a}_\not\perp, d)), (s_{\mathfrak{p}_\not\perp}, c, \lambda; s'_{\mathfrak{p}_\not\perp}, (\mathfrak{p}_\not\perp, d)) \mid c \in \Sigma_{\mathbf{c}} \\
&\quad \wedge\ (\exists\ s'' \mid (s'', \varepsilon, d; s', \perp) \in \delta : (s, c, \lambda; s'', d) \in \delta)\} \\
\cup\ &\{(s_{\mathfrak{a}_\not\perp}, r, (\mathfrak{a}_\perp, d); s'_{\mathfrak{a}_\perp}, \lambda), (s_{\mathfrak{a}_\not\perp}, r, (\mathfrak{a}_\not\perp, d); s'_{\mathfrak{a}_\not\perp}, \lambda), (s_{\mathfrak{a}_\not\perp}, r, (\mathfrak{p}_\not\perp, d); s'_{\mathfrak{p}_\not\perp}, \lambda) \mid r \in \Sigma_{\mathbf{r}} \\
&\quad \wedge\ (s, r, d; s', \lambda) \in \delta\} \\
\cup\ &\{(s_{\mathfrak{a}_\not\perp}, r, (\mathfrak{a}_\not\perp, d); s'_{\mathfrak{p}_\not\perp}, \lambda) \mid r \in \Sigma_{\mathbf{r}} \\
&\quad \wedge\ (\exists\ s'' \mid (\forall\ d' \mid d' \in \Gamma : (s'', \varepsilon, d'; s', \perp) \in \delta) : (s, r, d; s'', \lambda) \in \delta)\} \\
\cup\ &\{(s_{\mathfrak{a}_\perp}, r, \perp; s'_{\mathfrak{a}_\perp}, \perp) \mid r \in \Sigma_{\mathbf{r}} \wedge (s, r, \perp; s', \perp) \in \delta\} \\
\cup\ &\{(s_{\mathfrak{p}_\not\perp}, r, (\mathfrak{a}_\perp, d); s'_{\mathfrak{a}_\perp}, \lambda) \mid r \in \Sigma_{\mathbf{r}} \wedge d \in \Gamma \wedge (s, r, \perp; s', \perp) \in \delta\} \\
\cup\ &\{(s_{\mathfrak{p}_\not\perp}, r, (\mathfrak{a}_\not\perp, d); s'_{\mathfrak{p}_\not\perp}, \lambda), (s_{\mathfrak{p}_\not\perp}, r, (\mathfrak{p}_\not\perp, d); s'_{\mathfrak{p}_\not\perp}, \lambda) \mid r \in \Sigma_{\mathbf{r}} \wedge d \in \Gamma \\
&\quad \wedge\ (s, r, \perp; s', \perp) \in \delta\}\ .
\end{aligned}$$

The algebraic encoding of this automaton is

$$\left(\begin{bmatrix} \overrightarrow{I} \\ \hline \overrightarrow{0} \\ \hline \overrightarrow{0} \end{bmatrix}, \mathbf{WM}', \left[\begin{array}{c|c|c} \mathbf{0} & \mathbf{T_c} & \mathbf{T_c} \bullet \varepsilon_{\not\perp} \\ \hline \mathbf{0} & \mathbf{T_c} & \mathbf{T_c} \bullet \varepsilon_{\not\perp} \\ \hline \mathbf{0} & \mathbf{T_c} & \mathbf{T_c} \bullet \varepsilon_{\not\perp} \end{array}\right], \left[\begin{array}{c|c|c} \mathbf{T}_\perp & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array}\right], \mathbf{0}, \mathbf{0}, \begin{bmatrix} \overrightarrow{F} \\ \hline \overrightarrow{F} \\ \hline \overrightarrow{F} \end{bmatrix}\right)\ .$$

First, note that the list $\mathcal{B}'$ of $\mathbf{WM}'$ is defined by the following propositions:

a. $\left[_{s_{\mathfrak{a}_\perp}}\ 1\right]^{s_{\mathfrak{a}_\perp}} \in (\mathcal{B}')^1$, $\left[_{s_{\mathfrak{a}_\not\perp}}\ 1\right]^{s_{\mathfrak{a}_\not\perp}} \in (\mathcal{B}')^1$ and $\left[_{s_{\mathfrak{p}_\not\perp}}\ 1\right]^{s_{\mathfrak{p}_\not\perp}} \in (\mathcal{B}')^1$ for all $s \in S$;

b. $\left[_{s_{\mathfrak{a}_\perp}}\ a\right]^{s'_{\mathfrak{a}_\perp}} \in (\mathcal{B}')^1$, $\left[_{s_{\mathfrak{a}_\not\perp}}\ a\right]^{s'_{\mathfrak{a}_\not\perp}} \in (\mathcal{B}')^1$ and $\left[_{s_{\mathfrak{p}_\not\perp}}\ a\right]^{s'_{\mathfrak{p}_\not\perp}} \in (\mathcal{B}')^1$ for all $s, s' \in S$, $a \in \Sigma_{\mathbf{i}}$ and $\left[_{s}\ a\right]^{s'} \in \mathcal{B}^1$;

c. $\left[_{s_{\mathfrak{a}_\not\perp}}\ a\right]^{s'_{\mathfrak{p}_\not\perp}} \in (\mathcal{B}')^1$ for all $s, s' \in S$ and $a \in \Sigma_{\mathbf{i}}$ such that there exists a $s'' \in S$ such that $\vdash \varepsilon_{\not\perp}[s'', s'] = 1$ and $\left[_{s}\ a\right]^{s''} \in \mathcal{B}^1$;

d. $\left[_{s_{\mathfrak{a}_\perp}}\ c{\downarrow_{t_{\mathfrak{a}_\not\perp}}}{\uparrow^{t'_{\mathfrak{a}_\not\perp}}}\ r\right]^{s'_{\mathfrak{a}_\perp}} \in (\mathcal{B}')^2$, $\left[_{s_{\mathfrak{a}_\not\perp}}\ c{\downarrow_{t_{\mathfrak{a}_\not\perp}}}{\uparrow^{t'_{\mathfrak{a}_\not\perp}}}\ r\right]^{s'_{\mathfrak{a}_\not\perp}} \in (\mathcal{B}')^2$ and $\left[_{s_{\mathfrak{p}_\not\perp}}\ c{\downarrow_{t_{\mathfrak{a}_\not\perp}}}{\uparrow^{t'_{\mathfrak{a}_\not\perp}}}\ r\right]^{s'_{\mathfrak{p}_\not\perp}} \in (\mathcal{B}')^2$ for all $s, s', t, t' \in S$, $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$ such that $\left[_{s}\ c{\downarrow_{t}}{\uparrow^{t'}}\ r\right]^{s'} \in \mathcal{B}^2$;

e. $\left[_{s_{\mathfrak{a}_\perp}}\ c{\downarrow_{t_{\mathfrak{a}_\not\perp}}}{\uparrow^{t'_{\mathfrak{p}_\not\perp}}}\ r\right]^{s'_{\mathfrak{a}_\perp}} \in (\mathcal{B}')^2$ for all $s, s', t, t' \in S$, $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$ such that $\vdash \mathbf{T_c}[s, t] \geqslant c$ and $\vdash \mathbf{T}_\perp[t', s'] \geqslant r$;

f. $\lfloor_{s_{\mathfrak{a}_{\not\perp}}} c{\downarrow_{t_{\mathfrak{a}_{\not\perp}}}} \uparrow^{t'_{\mathfrak{p}_{\not\perp}}} r \rceil^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^2$ and $\lfloor_{s_{\mathfrak{p}_{\not\perp}}} c{\downarrow_{t_{\mathfrak{a}_{\not\perp}}}} \uparrow^{t'_{\mathfrak{p}_{\not\perp}}} r \rceil^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^2$ for all $s, s', t, t' \in S$, $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$ such that $\vdash \mathbf{T}_{\mathbf{c}}[s,t] \geqslant c$ and $\vdash \mathbf{T}_{\perp}[t',s'] \geqslant r$;

g. $\lfloor_{s_{\mathfrak{a}_{\not\perp}}} c{\downarrow_{t_{\mathfrak{a}_{\not\perp}}}} \uparrow^{t'_{\mathfrak{a}_{\not\perp}}} r \rceil^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^2$ for all $s, s', t, t' \in S$, $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$ such that there exists a $s'' \in S$ such that $\vdash \varepsilon_{\not\perp}[s'', s'] = 1$ and $\lfloor_s c{\downarrow_t} \uparrow^{t'} r \rceil^{s''} \in \mathcal{B}^2$;

h. $\lfloor_{s_{\mathfrak{a}_{\perp}}} c{\downarrow_{t_{\mathfrak{p}_{\not\perp}}}} \uparrow^{t'_{\mathfrak{a}_{\not\perp}}} r \rceil^{s'_{\mathfrak{a}_{\perp}}} \in (\mathcal{B}')^2$, $\lfloor_{s_{\mathfrak{a}_{\not\perp}}} c{\downarrow_{t_{\mathfrak{p}_{\not\perp}}}} \uparrow^{t'_{\mathfrak{a}_{\not\perp}}} r \rceil^{s'_{\mathfrak{a}_{\not\perp}}} \in (\mathcal{B}')^2$ and $\lfloor_{s_{\mathfrak{p}_{\not\perp}}} c{\downarrow_{t_{\mathfrak{p}_{\not\perp}}}} \uparrow^{t'_{\mathfrak{a}_{\not\perp}}} r \rceil^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^2$ for all $s, s', t, t' \in S$, $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$ such that there exists a $t'' \in S$ such that $\vdash \varepsilon_{\not\perp}[t'', t] = 1$ and $\lfloor_s c{\downarrow_{t''}} \uparrow^{t'} r \rceil^{s'} \in \mathcal{B}^2$;

i. $\lfloor_{s_{\mathfrak{a}_{\perp}}} c{\downarrow_{t_{\mathfrak{p}_{\not\perp}}}} \uparrow^{t'_{\mathfrak{p}_{\not\perp}}} r \rceil^{s'_{\mathfrak{a}_{\perp}}} \in (\mathcal{B}')^2$ for all $s, s', t, t' \in S$, $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$ such that there exists a $t'' \in S$ such that $\vdash \varepsilon_{\not\perp}[t'', t] = 1$, $\vdash \mathbf{T}_{\mathbf{c}}[s,t''] \geqslant c$ and $\vdash \mathbf{T}_{\perp}[t',s'] \geqslant r$;

j. $\lfloor_{s_{\mathfrak{a}_{\not\perp}}} c{\downarrow_{t_{\mathfrak{p}_{\not\perp}}}} \uparrow^{t'_{\mathfrak{p}_{\not\perp}}} r \rceil^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^2$ and $\lfloor_{s_{\mathfrak{p}_{\not\perp}}} c{\downarrow_{t_{\mathfrak{p}_{\not\perp}}}} \uparrow^{t'_{\mathfrak{p}_{\not\perp}}} r \rceil^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^2$ for all $s, s', t, t' \in S$, $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$ such that there exists a $t'' \in S$ such that $\vdash \varepsilon_{\not\perp}[t'', t] = 1$, $\vdash \mathbf{T}_{\mathbf{c}}[s,t''] \geqslant c$ and $\vdash \mathbf{T}_{\perp}[t',s'] \geqslant r$;

k. $\lfloor_{s_{\mathfrak{a}_{\not\perp}}} c{\downarrow_{t_{\mathfrak{p}_{\not\perp}}}} \uparrow^{t'_{\mathfrak{a}_{\not\perp}}} r \rceil^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^2$ for all $s, s', t, t' \in S$, $c \in \Sigma_{\mathbf{c}}$ and $r \in \Sigma_{\mathbf{r}}$ such that there exists $s'', t'' \in S$ such that $\vdash \varepsilon_{\not\perp}[s'', s'] = 1$, $\vdash \varepsilon_{\not\perp}[t'', t] = 1$ and $\lfloor_s c{\downarrow_{t''}} \uparrow^{t'} r \rceil^{s''} \in \mathcal{B}^2$.

Now, some results are given. Using these results, the proof of (C.31) is presented. Then, the remaining results are proved.

Note that

$$\vdash \mathbf{WM}' = \left[\begin{array}{c|c|c} \mathbf{A} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM} & \mathbf{B} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{C} \end{array}\right] , \tag{C.32}$$

where, for all $s, s' \in S$,

$$\mathbf{A}[s_{\mathfrak{a}_{\perp}}, s'_{\mathfrak{a}_{\perp}}] := \left(\!\!\left( \mathcal{B}' \right)\!\!\right)_{s_{\mathfrak{a}_{\perp}}}^{s'_{\mathfrak{a}_{\perp}}} ,$$

$$\mathbf{B}[s_{\mathfrak{a}_{\not\perp}}, s'_{\mathfrak{p}_{\not\perp}}] := \left(\!\!\left( \mathcal{B}' \right)\!\!\right)_{s_{\mathfrak{a}_{\not\perp}}}^{s'_{\mathfrak{p}_{\not\perp}}} ,$$

$$\mathbf{C}[s_{\mathfrak{p}_{\not\perp}}, s'_{\mathfrak{p}_{\not\perp}}] := \left(\!\!\left( \mathcal{B}' \right)\!\!\right)_{s_{\mathfrak{p}_{\not\perp}}}^{s'_{\mathfrak{p}_{\not\perp}}} .$$

Note also that the following equations hold:

$$\vdash \mathbf{A} = \mathbf{C} \ , \tag{C.33}$$

$$\begin{aligned} &\vdash (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_c \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star \\ &\quad = \ (\mathbf{T}_\perp + \mathbf{A})^\star \bullet ((\mathbf{T}_c \bullet \mathbf{WM})^\star \bullet (\mathbf{T}_c \bullet \mathbf{B} + \mathbf{T}_c \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \ . \end{aligned} \tag{C.34}$$

Using these results, we are able to prove (C.31).

$$\begin{aligned}
&\left[\begin{array}{c|c|c} \overrightarrow{I}^t & \overrightarrow{0} & \overrightarrow{0} \end{array}\right] \bullet \left( \left[\begin{array}{c|c|c} \mathbf{T}_\perp & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array}\right] + \mathbf{WM}' + \mathbf{0} \right. \\
&\qquad \left. + \left( \left[\begin{array}{c|c|c} \mathbf{0} & \mathbf{T}_c & \mathbf{T}_c \bullet \varepsilon_{\not\perp} \\ \hline \mathbf{0} & \mathbf{T}_c & \mathbf{T}_c \bullet \varepsilon_{\not\perp} \\ \hline \mathbf{0} & \mathbf{T}_c & \mathbf{T}_c \bullet \varepsilon_{\not\perp} \end{array}\right] \bullet \mathbf{WM}' \right)^+ \bullet \mathbf{0} \right)^\star \\
&\quad \bullet \left( \left[\begin{array}{c|c|c} \mathbf{0} & \mathbf{T}_c & \mathbf{T}_c \bullet \varepsilon_{\not\perp} \\ \hline \mathbf{0} & \mathbf{T}_c & \mathbf{T}_c \bullet \varepsilon_{\not\perp} \\ \hline \mathbf{0} & \mathbf{T}_c & \mathbf{T}_c \bullet \varepsilon_{\not\perp} \end{array}\right] \bullet \mathbf{WM}' \right)^\star \bullet \left[\begin{array}{c} \overrightarrow{F} \\ \hline \overrightarrow{F} \\ \hline \overrightarrow{F} \end{array}\right] \\
= \quad & \{\!\{ \text{ Zero of } \cdot \text{ \& Identity of } + \ \}\!\} \\
&\left[\begin{array}{c|c|c} \overrightarrow{I}^t & \overrightarrow{0} & \overrightarrow{0} \end{array}\right] \bullet \left( \left[\begin{array}{c|c|c} \mathbf{T}_\perp & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array}\right] + \mathbf{WM}' \right)^\star \\
&\quad \bullet \left( \left[\begin{array}{c|c|c} \mathbf{0} & \mathbf{T}_c & \mathbf{T}_c \bullet \varepsilon_{\not\perp} \\ \hline \mathbf{0} & \mathbf{T}_c & \mathbf{T}_c \bullet \varepsilon_{\not\perp} \\ \hline \mathbf{0} & \mathbf{T}_c & \mathbf{T}_c \bullet \varepsilon_{\not\perp} \end{array}\right] \bullet \mathbf{WM}' \right)^\star \bullet \left[\begin{array}{c} \overrightarrow{F} \\ \hline \overrightarrow{F} \\ \hline \overrightarrow{F} \end{array}\right] \\
= \quad & \{\!\{ \text{ Equation (C.32) } \}\!\} \\
&\left[\begin{array}{c|c|c} \overrightarrow{I}^t & \overrightarrow{0} & \overrightarrow{0} \end{array}\right] \bullet \left( \left[\begin{array}{c|c|c} \mathbf{T}_\perp & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array}\right] + \left[\begin{array}{c|c|c} \mathbf{A} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM} & \mathbf{B} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{C} \end{array}\right] \right)^\star \\
&\quad \bullet \left( \left[\begin{array}{c|c|c} \mathbf{0} & \mathbf{T}_c & \mathbf{T}_c \bullet \varepsilon_{\not\perp} \\ \hline \mathbf{0} & \mathbf{T}_c & \mathbf{T}_c \bullet \varepsilon_{\not\perp} \\ \hline \mathbf{0} & \mathbf{T}_c & \mathbf{T}_c \bullet \varepsilon_{\not\perp} \end{array}\right] \bullet \left[\begin{array}{c|c|c} \mathbf{A} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM} & \mathbf{B} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{C} \end{array}\right] \right)^\star \bullet \left[\begin{array}{c} \overrightarrow{F} \\ \hline \overrightarrow{F} \\ \hline \overrightarrow{F} \end{array}\right] \\
= \quad & \{\!\{ \text{ Definition of } \bullet \text{ and } + \text{ \& Kleene algebra } \}\!\} \\
&\left[\begin{array}{c|c|c} \overrightarrow{I}^t & \overrightarrow{0} & \overrightarrow{0} \end{array}\right] \bullet \left[\begin{array}{c|c|c} \mathbf{T}_\perp + \mathbf{A} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM} & \mathbf{B} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{C} \end{array}\right]^\star \\
&\quad \bullet \left[\begin{array}{c|c|c} \mathbf{0} & \mathbf{T}_c \bullet \mathbf{WM} & \mathbf{T}_c \bullet \mathbf{B} + \mathbf{T}_c \bullet \varepsilon_{\not\perp} \bullet \mathbf{C} \\ \hline \mathbf{0} & \mathbf{T}_c \bullet \mathbf{WM} & \mathbf{T}_c \bullet \mathbf{B} + \mathbf{T}_c \bullet \varepsilon_{\not\perp} \bullet \mathbf{C} \\ \hline \mathbf{0} & \mathbf{T}_c \bullet \mathbf{WM} & \mathbf{T}_c \bullet \mathbf{B} + \mathbf{T}_c \bullet \varepsilon_{\not\perp} \bullet \mathbf{C} \end{array}\right]^\star \bullet \left[\begin{array}{c} \overrightarrow{F} \\ \hline \overrightarrow{F} \\ \hline \overrightarrow{F} \end{array}\right]
\end{aligned}$$

$$
\begin{aligned}
&= \quad \{\!\{ \text{ Definition of } ^\star \text{ \& Kleene algebra } \}\!\} \\
&\left[\begin{array}{c|c|c} \overrightarrow{I}^t & \overrightarrow{0} & \overrightarrow{0} \end{array}\right] \bullet \left[\begin{array}{c|c|c} (\mathbf{T}_\perp + \mathbf{A})^\star & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM}^\star & \mathbf{WM}^\star \bullet \mathbf{B} \bullet \mathbf{C}^\star \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{C}^\star \end{array}\right] \\
&\bullet \left[\begin{array}{c|c|c} \mathbf{I} & ((\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{C})^\star \bullet \mathbf{T_c} \bullet \mathbf{WM})^+ & ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{C}))^+ \\ \hline \mathbf{0} & ((\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{C})^\star \bullet \mathbf{T_c} \bullet \mathbf{WM})^\star & ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{C}))^+ \\ \hline \mathbf{0} & ((\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{C})^\star \bullet \mathbf{T_c} \bullet \mathbf{WM})^+ & ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{C}))^\star \end{array}\right] \bullet \left[\begin{array}{c} \overrightarrow{F} \\ \hline \overrightarrow{F} \\ \hline \overrightarrow{F} \end{array}\right] \\
&= \quad \{\!\{ \text{ Definition of } \bullet \text{ \& Kleene algebra } \}\!\} \\
&\left[\begin{array}{c|c|c} \overrightarrow{I}^t \bullet (\mathbf{T}_\perp + \mathbf{A})^\star & \overrightarrow{0} & \overrightarrow{0} \end{array}\right] \bullet \left[\begin{array}{c} (\mathbf{T_c} \bullet \mathbf{WM} + \mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{C})^\star \bullet \overrightarrow{F} \\ \hline (\mathbf{T_c} \bullet \mathbf{WM} + \mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{C})^\star \bullet \overrightarrow{F} \\ \hline (\mathbf{T_c} \bullet \mathbf{WM} + \mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{C})^\star \bullet \overrightarrow{F} \end{array}\right] \\
&= \quad \{\!\{ \text{ Definition of } \bullet \text{ \& Kleene algebra } \}\!\} \\
&\overrightarrow{I}^t \bullet (\mathbf{T}_\perp + \mathbf{A})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM} + \mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{C})^\star \bullet \overrightarrow{F} \\
&= \quad \{\!\{ \text{ Denesting rule } \}\!\} \\
&\overrightarrow{I}^t \bullet (\mathbf{T}_\perp + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{C}))^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \\
&= \quad \{\!\{ \text{ Equation (C.33) } \}\!\} \\
&\overrightarrow{I}^t \bullet (\mathbf{T}_\perp + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \\
&= \quad \{\!\{ \text{ Equation (C.34) } \}\!\} \\
&\overrightarrow{I}^t \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \\
&= \quad \{\!\{ \text{ Hypothesis: } \mathcal{A} \text{ is a semi-visibly pushdown automaton such that } \delta \text{ does not contain } \varepsilon\text{-transition of the form } (s, \varepsilon, \perp; s', \perp) \text{ for all } s, s' \in S. \text{ So, } \varepsilon_\perp = \mathbf{0}. \}\!\} \\
&\overrightarrow{I}^t \bullet (\mathbf{T}_\perp + \mathbf{WM} + \varepsilon_\perp + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \qquad \blacksquare
\end{aligned}
$$

### C.2.1 Proof of (C.32)

**Proof of the case $\geqslant$ of (C.32):** All inequations are trivial except

$$\mathbf{WM}[s, s'] \leqslant \mathop{(\!|}_{s_{\mathfrak{a}_{\not\perp}}}^{s'_{\mathfrak{a}_{\not\perp}}} \mathcal{B}' \mathop{|\!)} \tag{C.35}$$

for every $s, s' \in S$. This inequation follows from (3.37) using the substitution function $g : S \to \{s_{\mathfrak{a}_\perp}, s_{\mathfrak{a}_{\not\perp}}, s_{\mathfrak{p}_{\not\perp}} \mid s \in S\}$ defined by $g(s) := s_{\mathfrak{a}_{\not\perp}}$ for all $s \in S$, and noting that $\widehat{g}(\mathcal{B})$ is a sublist of $\mathcal{B}'$.

**Proof of the case $\leqslant$ of (C.32):** We first show that

$$\vdash \mathbf{A} \leqslant \mathbf{C} \ . \tag{C.36}$$

This is direct from (3.37) using the substitution function $h : \{s_{\mathfrak{a}_\perp}, s_{\mathfrak{a}_\not\perp}, s_{\mathfrak{p}_\not\perp} \mid s \in S\} \to \{s_{\mathfrak{a}_\perp}, s_{\mathfrak{a}_\not\perp}, s_{\mathfrak{p}_\not\perp} \mid s \in S\}$ defined by

$$\begin{aligned} h(s_{\mathfrak{a}_\perp}) &:= s_{\mathfrak{p}_\not\perp} \ , \\ h(s_{\mathfrak{a}_\not\perp}) &:= s_{\mathfrak{a}_\not\perp} \ , \\ h(s_{\mathfrak{p}_\not\perp}) &:= s_{\mathfrak{p}_\not\perp} \ , \end{aligned}$$

for all $s \in S$, and noting that $\widehat{h}(\mathcal{B}')$ shrinked with idempotency of blocks is a sublist of $\mathcal{B}'$.

We now prove the case $\leqslant$ of (C.32). We use (3.5) with[1]

$$\begin{aligned} &s_{(s_{\mathfrak{a}_\perp}, s'_{\mathfrak{a}_\perp})} := (\!|_{s_{\mathfrak{a}_\perp}} \mathcal{B}' |\!)^{s'_{\mathfrak{a}_\perp}}, && s_{(s_{\mathfrak{a}_\perp}, s'_{\mathfrak{a}_\not\perp})} := 0, && s_{(s_{\mathfrak{a}_\perp}, s'_{\mathfrak{p}_\not\perp})} := 0, \\ &s_{(s_{\mathfrak{a}_\not\perp}, s'_{\mathfrak{a}_\perp})} := 0, && s_{(s_{\mathfrak{a}_\not\perp}, s'_{\mathfrak{a}_\not\perp})} := (\!|_s \mathcal{B} |\!)^{s'}, && s_{(s_{\mathfrak{a}_\not\perp}, s'_{\mathfrak{p}_\not\perp})} := (\!|_{s_{\mathfrak{a}_\not\perp}} \mathcal{B}' |\!)^{s'_{\mathfrak{p}_\not\perp}}, \\ &s_{(s_{\mathfrak{p}_\not\perp}, s'_{\mathfrak{a}_\perp})} := 0, && s_{(s_{\mathfrak{p}_\not\perp}, s'_{\mathfrak{a}_\not\perp})} := 0, && s_{(s_{\mathfrak{p}_\not\perp}, s'_{\mathfrak{p}_\not\perp})} := (\!|_{s_{\mathfrak{a}_\perp}} \mathcal{B}' |\!)^{s'_{\mathfrak{a}_\perp}} . \end{aligned}$$

So, noting that each block of $\mathcal{B}'$ having as starting label a $t_{\mathfrak{a}_\perp}$ (respectively, $t_{\mathfrak{p}_\not\perp}$) must also have as ending label a $t'_{\mathfrak{a}_\perp}$ (respectively, $t'_{\mathfrak{p}_\not\perp}$) where $t, t' \in S$, and each block of $\mathcal{B}'$ having as starting label a $t_{\mathfrak{a}_\not\perp}$ must also have as ending label a $t'_{\mathfrak{a}_\not\perp}$ or $t'_{\mathfrak{p}_\not\perp}$ where $t, t' \in S$, it suffices to prove, for all $s, s' \in S$,

$$(\wedge\, m \mid [\!]_{s_{\mathfrak{a}_\perp}}^{s'_{\mathfrak{a}_\perp}} m \in (\mathcal{B}')^1 : m \leqslant (\!|_{s_{\mathfrak{a}_\perp}}^{s'_{\mathfrak{a}_\perp}} \mathcal{B}' |\!)) \ , \tag{C.37}$$

$$(\wedge\, m \mid [\!]_{s_{\mathfrak{a}_\not\perp}}^{s'_{\mathfrak{a}_\not\perp}} m \in (\mathcal{B}')^1 : m \leqslant (\!|_{s}^{s'} \mathcal{B} |\!)) \ , \tag{C.38}$$

$$(\wedge\, m \mid [\!]_{s_{\mathfrak{a}_\not\perp}}^{s'_{\mathfrak{p}_\not\perp}} m \in (\mathcal{B}')^1 : m \leqslant (\!|_{s_{\mathfrak{a}_\not\perp}}^{s'_{\mathfrak{p}_\not\perp}} \mathcal{B}' |\!)) \ , \tag{C.39}$$

$$(\wedge\, m \mid [\!]_{s_{\mathfrak{p}_\not\perp}}^{s'_{\mathfrak{p}_\not\perp}} m \in (\mathcal{B}')^1 : m \leqslant (\!|_{s_{\mathfrak{a}_\perp}}^{s'_{\mathfrak{a}_\perp}} \mathcal{B}' |\!)) \ , \tag{C.40}$$

$$(\wedge\, c, z, r, w \mid [\!]_{s_{\mathfrak{a}_\perp}}^{s'_{\mathfrak{a}_\perp}} c{\downarrow_z^w}{\uparrow} r \in (\mathcal{B}')^2 : c \cdot s_{(z,w)} \cdot r \leqslant (\!|_{s_{\mathfrak{a}_\perp}}^{s'_{\mathfrak{a}_\perp}} \mathcal{B}' |\!)) \ , \tag{C.41}$$

$$(\wedge\, c, z, r, w \mid [\!]_{s_{\mathfrak{a}_\not\perp}}^{s'_{\mathfrak{a}_\not\perp}} c{\downarrow_z^w}{\uparrow} r \in (\mathcal{B}')^2 : c \cdot s_{(z,w)} \cdot r \leqslant (\!|_{s}^{s'} \mathcal{B} |\!)) \ , \tag{C.42}$$

[1]Note that we defined $s_{(s_{\mathfrak{p}_\not\perp}, s'_{\mathfrak{p}_\not\perp})} := (\!|_{s_{\mathfrak{a}_\perp}} \mathcal{B}' |\!)^{s'_{\mathfrak{a}_\perp}}$ instead of $s_{(s_{\mathfrak{p}_\not\perp}, s'_{\mathfrak{p}_\not\perp})} := (\!|_{s_{\mathfrak{p}_\not\perp}} \mathcal{B}' |\!)^{s'_{\mathfrak{p}_\not\perp}}$. This is sufficient by (C.36) and Kleene algebra. This particular solution is used to ease the proof of (C.33).

$$(\wedge\, c, z, r, w \mid {\scriptstyle s_{\mathfrak{a}_{\not\perp}}}\big[\; c{\downarrow_z}{\overset{w}{\uparrow}}r \;\big]^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^2 : c \cdot s_{(z,w)} \cdot r \leqslant {\scriptstyle s_{\mathfrak{a}_{\not\perp}}}(\!|\, \mathcal{B}' \,|\!)^{s'_{\mathfrak{p}_{\not\perp}}}) \ , \tag{C.43}$$

$$(\wedge\, c, z, r, w \mid {\scriptstyle s_{\mathfrak{p}_{\not\perp}}}\big[\; c{\downarrow_z}{\overset{w}{\uparrow}}r \;\big]^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^2 : c \cdot s_{(z,w)} \cdot r \leqslant {\scriptstyle s_{\mathfrak{a}_\perp}}(\!|\, \mathcal{B}' \,|\!)^{s'_{\mathfrak{a}_\perp}}) \ , \tag{C.44}$$

$$(\wedge\, v \mid v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\} : s_{(s_{\mathfrak{a}_\perp}, v)} \cdot s_{(v, s'_{\mathfrak{a}_\perp})} \leqslant {\scriptstyle s_{\mathfrak{a}_\perp}}(\!|\, \mathcal{B}' \,|\!)^{s'_{\mathfrak{a}_\perp}}) \ , \tag{C.45}$$

$$(\wedge\, v \mid v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\} : s_{(s_{\mathfrak{a}_\perp}, v)} \cdot s_{(v, s'_{\mathfrak{a}_{\not\perp}})} \leqslant 0) \ , \tag{C.46}$$

$$(\wedge\, v \mid v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\} : s_{(s_{\mathfrak{a}_\perp}, v)} \cdot s_{(v, s'_{\mathfrak{p}_{\not\perp}})} \leqslant 0) \ , \tag{C.47}$$

$$(\wedge\, v \mid v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\} : s_{(s_{\mathfrak{a}_{\not\perp}}, v)} \cdot s_{(v, s'_{\mathfrak{a}_\perp})} \leqslant 0) \ , \tag{C.48}$$

$$(\wedge\, v \mid v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\} : s_{(s_{\mathfrak{a}_{\not\perp}}, v)} \cdot s_{(v, s'_{\mathfrak{a}_{\not\perp}})} \leqslant {\scriptstyle s}(\!|\, \mathcal{B} \,|\!)^{s'}) \ , \tag{C.49}$$

$$(\wedge\, v \mid v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\} : s_{(s_{\mathfrak{a}_{\not\perp}}, v)} \cdot s_{(v, s'_{\mathfrak{p}_{\not\perp}})} \leqslant {\scriptstyle s_{\mathfrak{a}_{\not\perp}}}(\!|\, \mathcal{B}' \,|\!)^{s'_{\mathfrak{p}_{\not\perp}}}) \ , \tag{C.50}$$

$$(\wedge\, v \mid v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\} : s_{(s_{\mathfrak{p}_{\not\perp}}, v)} \cdot s_{(v, s'_{\mathfrak{a}_\perp})} \leqslant 0) \ , \tag{C.51}$$

$$(\wedge\, v \mid v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\} : s_{(s_{\mathfrak{p}_{\not\perp}}, v)} \cdot s_{(v, s'_{\mathfrak{a}_{\not\perp}})} \leqslant 0) \ , \tag{C.52}$$

$$(\wedge\, v \mid v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\} : s_{(s_{\mathfrak{p}_{\not\perp}}, v)} \cdot s_{(v, s'_{\mathfrak{p}_{\not\perp}})} \leqslant {\scriptstyle s_{\mathfrak{a}_\perp}}(\!|\, \mathcal{B}' \,|\!)^{s'_{\mathfrak{a}_\perp}}) \ . \tag{C.53}$$

The proofs of inequations (C.37) are direct by axiom (3.2).

For inequations (C.38), we suppose ${\scriptstyle s_{\mathfrak{a}_{\not\perp}}}[\, m \,]^{s'_{\mathfrak{a}_{\not\perp}}} \in (\mathcal{B}')^1$ and we prove

$$m \leqslant {\scriptstyle s}(\!|\, \mathcal{B} \,|\!)^{s'} \ .$$

By the definition of $\mathcal{B}'$, conditions a and b, and the definition of $\mathcal{B}$, ${\scriptstyle s}[\, m \,]^{s'} \in \mathcal{B}^1$. So, the proof follows from axiom (3.2).

The proofs of inequations (C.39) are direct by axiom (3.2).

For inequations (C.40), we suppose ${\scriptstyle s_{\mathfrak{p}_{\not\perp}}}[\, m \,]^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^1$ and we prove

$$m \leqslant {\scriptstyle s_{\mathfrak{a}_\perp}}(\!|\, \mathcal{B}' \,|\!)^{s'_{\mathfrak{a}_\perp}} \ .$$

By the definition of $\mathcal{B}'$, conditions a and b, ${\scriptstyle s_{\mathfrak{a}_\perp}}[\, m \,]^{s'_{\mathfrak{a}_\perp}} \in (\mathcal{B}')^1$. So, the proof follows from axiom (3.2).

For inequations (C.41), we suppose $\lfloor_{s_{\mathfrak{a}_\perp}} c{\downarrow_z}{\uparrow^w} r \rceil^{s'_{\mathfrak{a}_\perp}} \in (\mathcal{B}')^2$ and we prove

$$c \cdot s_{(z,w)} \cdot r \leqslant \big(\!\big|_{s_{\mathfrak{a}_\perp}} \mathcal{B}' \big|\!\big)^{s'_{\mathfrak{a}_\perp}} \quad . \tag{C.54}$$

First, let us prove that

$$s_{(z,w)} \leqslant \big(\!\big|_{z} \mathcal{B}' \big|\!\big)^{w} \quad . \tag{C.55}$$

By the definition of $\mathcal{B}'$, conditions d, e, h and i, $z \in \{z'_{\mathfrak{a}_\not\perp}, z'_{\mathfrak{p}_\not\perp} \mid z' \in S\}$ and $w \in \{w'_{\mathfrak{a}_\not\perp}, w'_{\mathfrak{p}_\not\perp} \mid w' \in S\}$. If $z = z'_{\mathfrak{a}_\not\perp}$ for a $z' \in S$ and $w = w'_{\mathfrak{a}_\not\perp}$ for a $w' \in S$, then $s_{(z'_{\mathfrak{a}_\not\perp}, w'_{\mathfrak{a}_\not\perp})} = \big(\!\big|_{z'} \mathcal{B} \big|\!\big)^{w'}$. The proof of (C.55) follows from (C.35). If $z = z'_{\mathfrak{a}_\not\perp}$ for a $z' \in S$ and $w = w'_{\mathfrak{p}_\not\perp}$ for a $w' \in S$, then $s_{(z'_{\mathfrak{a}_\not\perp}, w'_{\mathfrak{p}_\not\perp})} = \big(\!\big|_{z'_{\mathfrak{a}_\not\perp}} \mathcal{B}' \big|\!\big)^{w'_{\mathfrak{p}_\not\perp}}$. The proof of (C.55) follows from reflexivity of $\leqslant$. If $z = z'_{\mathfrak{p}_\not\perp}$ for a $z' \in S$ and $w = w'_{\mathfrak{a}_\not\perp}$ for a $w' \in S$, then $s_{(z'_{\mathfrak{p}_\not\perp}, w'_{\mathfrak{a}_\not\perp})} = 0$. The proof of (C.55) follows from the fact that 0 is the minimum of the algebra. If $z = z'_{\mathfrak{p}_\not\perp}$ for a $z' \in S$ and $w = w'_{\mathfrak{p}_\not\perp}$ for a $w' \in S$, then $s_{(z'_{\mathfrak{p}_\not\perp}, w'_{\mathfrak{p}_\not\perp})} = \big(\!\big|_{z'_{\mathfrak{a}_\perp}} \mathcal{B}' \big|\!\big)^{w'_{\mathfrak{a}_\perp}}$. The proof of (C.55) follows from (C.36).

We now prove (C.54). The proof follows from (C.55), Kleene algebra and axiom (3.3).

For inequations (C.42), we suppose $\lfloor_{s_{\mathfrak{a}_\not\perp}} c{\downarrow_z}{\uparrow^w} r \rceil^{s'_{\mathfrak{a}_\not\perp}} \in (\mathcal{B}')^2$ and we prove

$$c \cdot s_{(z,w)} \cdot r \leqslant \big(\!\big|_{s} \mathcal{B} \big|\!\big)^{s'} \quad . \tag{C.56}$$

By the definition of $\mathcal{B}'$, conditions d and h, $z \in \{z'_{\mathfrak{a}_\not\perp}, z'_{\mathfrak{p}_\not\perp} \mid z' \in S\}$ and $w \in \{w'_{\mathfrak{a}_\not\perp} \mid w' \in S\}$. If $z = z'_{\mathfrak{a}_\not\perp}$ for a $z' \in S$ and $w = w'_{\mathfrak{a}_\not\perp}$ for a $w' \in S$, then $s_{(z'_{\mathfrak{a}_\not\perp}, w'_{\mathfrak{a}_\not\perp})} = \big(\!\big|_{z'} \mathcal{B} \big|\!\big)^{w'}$ and $\lfloor_s c{\downarrow_{z'}}{\uparrow^{w'}} r \rceil^{s'} \in \mathcal{B}^2$. So, the proof of (C.56) follows from (3.3). If $z = z'_{\mathfrak{p}_\not\perp}$ for a $z' \in S$ and $w = w'_{\mathfrak{a}_\not\perp}$ for a $w' \in S$, then $s_{(z'_{\mathfrak{p}_\not\perp}, w'_{\mathfrak{a}_\not\perp})} = 0$. The proof of (C.56) follows from zero of $\cdot$ and the fact that 0 is the minimum of the algebra.

For inequations (C.43), we suppose $\lfloor_{s_{\mathfrak{a}_\not\perp}} c{\downarrow_z}{\uparrow^w} r \rceil^{s'_{\mathfrak{p}_\not\perp}} \in (\mathcal{B}')^2$ and we prove

$$c \cdot s_{(z,w)} \cdot r \leqslant \big(\!\big|_{s_{\mathfrak{a}_\not\perp}} \mathcal{B}' \big|\!\big)^{s'_{\mathfrak{p}_\not\perp}} \quad . \tag{C.57}$$

First, let us prove that

$$s_{(z,w)} \leqslant \big(\!\big|_{z} \mathcal{B}' \big|\!\big)^{w} \quad . \tag{C.58}$$

By the definition of $\mathcal{B}'$, conditions f, g, j and k, $z \in \{z'_{\mathfrak{a}_\not\perp}, z'_{\mathfrak{p}_\not\perp} \mid z' \in S\}$ and $w \in \{w'_{\mathfrak{a}_\not\perp}, w'_{\mathfrak{p}_\not\perp} \mid w' \in S\}$. If $z = z'_{\mathfrak{a}_\not\perp}$ for a $z' \in S$ and $w = w'_{\mathfrak{a}_\not\perp}$ for a $w' \in S$, then

$s_{(z'_{\mathfrak{a}_\not},w'_{\mathfrak{a}_\not})} = (\!|_{z'}\, \mathcal{B}\, |\!)^{w'}$. The proof of (C.58) follows from (C.35). If $z = z'_{\mathfrak{a}_\not}$ for a $z' \in S$ and $w = w'_{\mathfrak{p}_\not}$ for a $w' \in S$, then $s_{(z'_{\mathfrak{a}_\not},w'_{\mathfrak{p}_\not})} = (\!|_{z'_{\mathfrak{a}_\not}}\, \mathcal{B}'\, |\!)^{w'_{\mathfrak{p}_\not}}$. The proof of (C.58) follows from reflexivity of $\leqslant$. If $z = z'_{\mathfrak{p}_\not}$ for a $z' \in S$ and $w = w'_{\mathfrak{a}_\not}$ for a $w' \in S$, then $s_{(z'_{\mathfrak{p}_\not},w'_{\mathfrak{a}_\not})} = 0$. The proof of (C.58) follows from the fact that 0 is the minimum of the algebra. If $z = z'_{\mathfrak{p}_\not}$ for a $z' \in S$ and $w = w'_{\mathfrak{p}_\not}$ for a $w' \in S$, then $s_{(z'_{\mathfrak{p}_\not},w'_{\mathfrak{p}_\not})} = (\!|_{z'_{\mathfrak{a}_\perp}}\, \mathcal{B}'\, |\!)^{w'_{\mathfrak{a}_\perp}}$. The proof of (C.58) follows from (C.36).

We now prove (C.57). The proof follows from (C.58), Kleene algebra and axiom (3.3).

For inequations (C.44), we suppose $[_{s_{\mathfrak{p}_\not}}\, c{\downarrow_z}{\uparrow^w}\, r\, ]^{s'_{\mathfrak{p}_\not}} \in (\mathcal{B}')^2$ and we prove

$$c \cdot s_{(z,w)} \cdot r \leqslant \underset{s_{\mathfrak{a}_\perp}}{(\!|}\, \mathcal{B}'\, \overset{s'_{\mathfrak{a}_\perp}}{|\!)} \quad .$$

By the definition of $\mathcal{B}'$, conditions d, e, f, h, i and j,

$$\underset{s_{\mathfrak{p}_\not}}{[}\ c\underset{z}{\downarrow}\overset{w}{\uparrow} r\ \overset{s'_{\mathfrak{p}_\not}}{]} \in (\mathcal{B}')^2 \Leftrightarrow \underset{s_{\mathfrak{a}_\perp}}{[}\ c\underset{z}{\downarrow}\overset{w}{\uparrow} r\ \overset{s'_{\mathfrak{a}_\perp}}{]} \in (\mathcal{B}')^2 \quad .$$

So, the proof follows from (C.41).

For inequations (C.45), there are two possible cases for $v$. If $v = v'_{\mathfrak{a}_\perp}$ for a $v' \in S$, then it must be proved that

$$\underset{s_{\mathfrak{a}_\perp}}{(\!|}\, \mathcal{B}'\, \overset{v'_{\mathfrak{a}_\perp}}{|\!)} \cdot \underset{v'_{\mathfrak{a}_\perp}}{(\!|}\, \mathcal{B}'\, \overset{s'_{\mathfrak{a}_\perp}}{|\!)} \leqslant \underset{s_{\mathfrak{a}_\perp}}{(\!|}\, \mathcal{B}'\, \overset{s'_{\mathfrak{a}_\perp}}{|\!)} \quad .$$

This is direct by axiom (3.4). If $v \in \{v'_{\mathfrak{a}_\not}, v'_{\mathfrak{p}_\not} \mid v' \in S\}$, then it must be proved that

$$0 \cdot s_{(v,s'_{\mathfrak{a}_\perp})} \leqslant \underset{s_{\mathfrak{a}_\perp}}{(\!|}\, \mathcal{B}'\, \overset{s'_{\mathfrak{a}_\perp}}{|\!)} \quad .$$

This is direct by zero of $\cdot$ and the fact that 0 is the minimum element of the algebra.

For inequations (C.46), there are two possible cases for $v$. If $v = v'_{\mathfrak{a}_\perp}$ for a $v' \in S$, then it must be proved that

$$\underset{s_{\mathfrak{a}_\perp}}{(\!|}\, \mathcal{B}'\, \overset{v'_{\mathfrak{a}_\perp}}{|\!)} \cdot 0 \leqslant 0 \quad .$$

This is direct by zero of $\cdot$ and the reflexivity of $\leqslant$. If $v \in \{v'_{\mathfrak{a}_\not}, v'_{\mathfrak{p}_\not} \mid v' \in S\}$, then it must be proved that

$$0 \cdot s_{(v,s'_{\mathfrak{a}_\not})} \leqslant 0 \quad .$$

This is direct by zero of $\cdot$ and the reflexivity of $\leqslant$.

The proofs of (C.47) are similar to the proofs of (C.46).

For inequations (C.48), there are two possible cases for $v$. If $v = v'_{\mathfrak{a}_\perp}$ for a $v' \in S$, then it must be proved that

$$0 \cdot \underset{v'_{\mathfrak{a}_\perp}}{(\!|\, \mathcal{B}' \,|\!)}^{s'_{\mathfrak{a}_\perp}} \leqslant 0 \ .$$

This is direct by zero of $\cdot$ and the reflexivity of $\leqslant$. If $v \in \{v'_{\mathfrak{a}_\not\perp}, v'_{\mathfrak{p}_\not\perp} \mid v' \in S\}$, then it must be proved that

$$s_{(s_{\mathfrak{a}_\not\perp}, v)} \cdot 0 \leqslant 0 \ .$$

This is direct by zero of $\cdot$ and the reflexivity of $\leqslant$.

For inequations (C.49), there are two possible cases for $v$. If $v = v'_{\mathfrak{a}_\not\perp}$ for a $v' \in S$, then it must be proved that

$$\underset{s}{(\!|\, \mathcal{B} \,|\!)}^{v'} \cdot \underset{v'}{(\!|\, \mathcal{B} \,|\!)}^{s'} \leqslant \underset{s}{(\!|\, \mathcal{B} \,|\!)}^{s'} \ .$$

This is direct by axiom (3.4). If $v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{p}_\not\perp} \mid v' \in S\}$, then it must be proved that

$$s_{(s_{\mathfrak{a}_\not\perp}, v)} \cdot 0 \leqslant \underset{s}{(\!|\, \mathcal{B} \,|\!)}^{s'} \ .$$

This is direct by zero of $\cdot$ and the fact that 0 is the minimum element of the algebra.

For inequations (C.50), there are three possible cases for $v$. If $v = v'_{\mathfrak{a}_\perp}$ for a $v' \in S$, then it must be proved that

$$0 \cdot 0 \leqslant \underset{s_{\mathfrak{a}_\not\perp}}{(\!|\, \mathcal{B}' \,|\!)}^{s'_{\mathfrak{p}_\not\perp}} \ .$$

This is direct by zero of $\cdot$ and the fact that 0 is the minimum element of the algebra. If $v = v'_{\mathfrak{a}_\not\perp}$ for a $v' \in S$, then it must be proved that

$$\underset{s}{(\!|\, \mathcal{B} \,|\!)}^{v'} \cdot \underset{v'_{\mathfrak{a}_\not\perp}}{(\!|\, \mathcal{B}' \,|\!)}^{s'_{\mathfrak{p}_\not\perp}} \leqslant \underset{s_{\mathfrak{a}_\not\perp}}{(\!|\, \mathcal{B}' \,|\!)}^{s'_{\mathfrak{p}_\not\perp}} \ .$$

This is direct by (C.35), Kleene algebra and axiom (3.4). If $v = v'_{\mathfrak{p}_\not\perp}$ for a $v' \in S$, then it must be proved that

$$\underset{s_{\mathfrak{a}_\not\perp}}{(\!|\, \mathcal{B}' \,|\!)}^{v'_{\mathfrak{p}_\not\perp}} \cdot \underset{v'_{\mathfrak{a}_\perp}}{(\!|\, \mathcal{B}' \,|\!)}^{s'_{\mathfrak{a}_\perp}} \leqslant \underset{s_{\mathfrak{a}_\not\perp}}{(\!|\, \mathcal{B}' \,|\!)}^{s'_{\mathfrak{p}_\not\perp}} \ .$$

This is direct by (C.36), Kleene algebra and axiom (3.4).

The proofs of (C.51) and (C.52) are similar to the proofs of (C.46).

For inequations (C.53), there are two possible cases for $v$. If $v = v'_{\mathfrak{p}_{\not\perp}}$ for a $v' \in S$, then it must be proved that

$$\left(\!\!\left|\, \mathcal{B}' \,\right|\!\!\right)^{v'_{\mathfrak{a}_\perp}}_{s_{\mathfrak{a}_\perp}} \cdot \left(\!\!\left|\, \mathcal{B}' \,\right|\!\!\right)^{s'_{\mathfrak{a}_\perp}}_{v'_{\mathfrak{a}_\perp}} \leqslant \left(\!\!\left|\, \mathcal{B}' \,\right|\!\!\right)^{s'_{\mathfrak{a}_\perp}}_{s_{\mathfrak{a}_\perp}} \quad .$$

This is direct by axiom (3.4). If $v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}} \mid v' \in S\}$, then it must be proved that

$$0 \cdot s_{(v, s'_{\mathfrak{p}_{\not\perp}})} \leqslant \left(\!\!\left|\, \mathcal{B}' \,\right|\!\!\right)^{s'_{\mathfrak{a}_\perp}}_{s_{\mathfrak{a}_\perp}} \quad .$$

This is direct by zero of $\cdot$ and the fact that 0 is the minimum element of the algebra.

### C.2.2 Proof of (C.33)

Equation (C.33) is shown in the case $\leqslant$ of the proof of (C.32). The case $\leqslant$ is exactly (C.36). The case $\geqslant$ is showed in page 222 when using the axiom (3.5) with solutions $s_{(s_{\mathfrak{p}_{\not\perp}}, s'_{\mathfrak{p}_{\not\perp}})} := \left(\!\!\left|\, \mathcal{B}' \,\right|\!\!\right)^{s'_{\mathfrak{a}_\perp}}_{s_{\mathfrak{a}_\perp}}$ for every $s, s' \in S$. This states that $\left(\!\!\left|\, \mathcal{B}' \,\right|\!\!\right)^{s'_{\mathfrak{p}_{\not\perp}}}_{s_{\mathfrak{p}_{\not\perp}}} \leqslant \left(\!\!\left|\, \mathcal{B}' \,\right|\!\!\right)^{s'_{\mathfrak{a}_\perp}}_{s_{\mathfrak{a}_\perp}}$ for every $s, s' \in S$.

### C.2.3 Proof of (C.34)

First, some results are given. Using these results, the proof of (C.34) is presented. Then, the remaining results are proved.

Note that the following equations and inequations hold:

$$\vdash \mathbf{WM} \leqslant \mathbf{A} \ , \tag{C.59}$$

$$\vdash \mathbf{A} \bullet \mathbf{A} = \mathbf{A} \ , \tag{C.60}$$

$$\vdash \mathbf{A}^\star = \mathbf{A} \ , \tag{C.61}$$

$$\vdash \mathbf{WM} \bullet \mathbf{B} = \mathbf{B} \ , \tag{C.62}$$

$$\vdash \mathbf{B} \bullet \mathbf{C} = \mathbf{B} \ , \tag{C.63}$$

$$\vdash \mathbf{WM} \bullet \varepsilon_{\not L} \leqslant \mathbf{B} + \varepsilon_{\not L} \ , \tag{C.64}$$
$$\vdash \mathbf{T_c} \bullet \mathbf{B} \bullet \mathbf{T}_\perp \leqslant \mathbf{A} \ , \tag{C.65}$$
$$\vdash \mathbf{T_c} \bullet \mathbf{B} \bullet \mathbf{T}_\perp \leqslant \mathbf{B} \ , \tag{C.66}$$
$$\vdash \mathbf{T_c} \bullet \varepsilon_{\not L} \bullet \mathbf{A} \bullet \mathbf{T}_\perp \leqslant \mathbf{A} \ , \tag{C.67}$$
$$\vdash \mathbf{T_c} \bullet \varepsilon_{\not L} \bullet \mathbf{A} \bullet \mathbf{T}_\perp \leqslant \mathbf{B} \ , \tag{C.68}$$
$$\vdash \mathbf{A} \leqslant (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not L})^\star \ , \tag{C.69}$$
$$\vdash \mathbf{B} \leqslant (\mathbf{WM} + \mathbf{T_c})^\star \bullet \varepsilon_{\not L} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not L})^\star \ . \tag{C.70}$$

We now prove (C.34).

**Proof of the case $\geqslant$ of (C.34):** We are able to prove it.

$$\begin{aligned}
& (\mathbf{T}_\perp + \mathbf{A})^\star \bullet \Big( (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not L} \bullet \mathbf{A}) \Big)^\star \\
\leqslant \quad & \text{\{\!\{ Inequations (C.69) and (C.70) \& Monotonicity of } \cdot \text{ and } + \text{ \}\!\}} \\
& \Big( \mathbf{T}_\perp + (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not L})^\star \Big)^\star \\
& \bullet \Big( (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \Big( \mathbf{T_c} \bullet (\mathbf{WM} + \mathbf{T_c})^\star \bullet \varepsilon_{\not L} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not L})^\star \\
& \qquad + \mathbf{T_c} \bullet \varepsilon_{\not L} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not L})^\star \Big) \Big)^\star \\
= \quad & \text{\{\!\{ Kleene algebra: } \mathbf{T}_\perp \leqslant (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not L})^\star \text{, definition of} \\
& \quad \leqslant \text{ and } (p^*)^* = p^* \text{ \& Identity of } \cdot \text{ \}\!\}} \\
& (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not L})^\star \\
& \bullet \Big( (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \Big( \mathbf{T_c} \bullet (\mathbf{WM} + \mathbf{T_c})^\star \bullet \varepsilon_{\not L} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not L})^\star \\
& \qquad + \mathbf{T_c} \bullet \mathbf{I} \bullet \varepsilon_{\not L} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not L})^\star \Big) \Big)^\star \\
\leqslant \quad & \text{\{\!\{ Denesting rule \& Lemma 4.5, properties i and iii \& Kleene algebra \}\!\}} \\
& (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not L})^\star \\
& \bullet \Big( (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \mathbf{T_c} \bullet \mathbf{WM} \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \varepsilon_{\not L} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not L})^\star \\
& \quad + (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \mathbf{T_c} \bullet \mathbf{WM} \bullet \varepsilon_{\not L} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not L})^\star \Big)^\star \\
= \quad & \text{\{\!\{ Sliding rule \& Kleene algebra: } p^* p^* = p^* \text{ \& Definition of } {}^+ \text{ \}\!\}} \\
& (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not L})^\star \\
& \bullet \Big( (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not L} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not L})^\star \\
& \quad + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not L} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not L})^\star \Big)^\star \\
\leqslant \quad & \text{\{\!\{ Kleene algebra (including idempotency of } +) \text{ \}\!\}}
\end{aligned}$$

$$
\begin{array}{ll}
& (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star \\
& \bullet \left((\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star\right)^\star \\
= & \qquad \{\!\{ \text{ Kleene algebra: } (p^*)^* = p^* \text{ and } p^* p^* = p^* \ \}\!\} \\
& (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star
\end{array}
$$

**Proof of the case $\leqslant$ of (C.34):** Using $^\star$-induction and Kleene algebra, it suffices to show that

$$
\begin{array}{ll}
\vdash & \mathbf{I} \\
\leqslant & (\mathbf{T}_\perp + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \ , \qquad \text{(C.71)} \\
\vdash & (\mathbf{T}_\perp + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \bullet \mathbf{T}_\perp \\
\leqslant & (\mathbf{T}_\perp + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \ , \qquad \text{(C.72)} \\
\vdash & (\mathbf{T}_\perp + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \bullet \mathbf{WM} \\
\leqslant & (\mathbf{T}_\perp + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \ , \qquad \text{(C.73)} \\
\vdash & (\mathbf{T}_\perp + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \\
& \quad \bullet \ (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp} \\
\leqslant & (\mathbf{T}_\perp + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \ . \qquad \text{(C.74)}
\end{array}
$$

Inequation (C.71) is direct from Kleene algebra.

For inequation (C.72),

$$
\begin{array}{ll}
& (\mathbf{T}_\perp + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \bullet \mathbf{T}_\perp \\
= & \qquad \{\!\{ \text{ Kleene algebra } \}\!\} \\
& (\mathbf{T}_\perp + \mathbf{A})^\star \bullet \mathbf{T}_\perp \\
& + (\mathbf{T}_\perp + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} \bullet \\
& \mathbf{T}_\perp + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A} \bullet \mathbf{T}_\perp) \\
\leqslant & \qquad \{\!\{ \text{ Kleene algebra } \}\!\} \\
& (\mathbf{T}_\perp + \mathbf{A})^\star \bullet (\mathbf{T}_\perp + \mathbf{A})^\star \\
& + (\mathbf{T}_\perp + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} \bullet \mathbf{T}_\perp + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A} \bullet \mathbf{T}_\perp) \\
& + (\mathbf{T}_\perp + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \mathbf{T_c} \bullet \mathbf{WM} \bullet \\
& (\mathbf{T_c} \bullet \mathbf{B} \bullet \mathbf{T}_\perp + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A} \bullet \mathbf{T}_\perp) \\
\leqslant & \qquad \{\!\{ \text{ Inequations (C.65), (C.66), (C.67) and (C.68) \& Kleene algebra } \}\!\} \\
& (\mathbf{T}_\perp + \mathbf{A})^\star \\
& + (\mathbf{T}_\perp + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \bullet \mathbf{A} \\
& + (\mathbf{T}_\perp + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \mathbf{T_c} \bullet \mathbf{WM} \bullet \mathbf{B} \\
\leqslant & \qquad \{\!\{ \text{ Equation (C.62) \& Kleene algebra } \}\!\}
\end{array}
$$

$$
\begin{aligned}
& (\mathbf{T}_\bot + \mathbf{A})^\star \\
& + (\mathbf{T}_\bot + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \bullet \mathbf{A} \\
& + (\mathbf{T}_\bot + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \\
= \quad & \{\!\{ \text{ Kleene algebra \& Equations (C.63), (C.33) and (C.60) } \}\!\} \\
& (\mathbf{T}_\bot + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} \bullet \mathbf{A} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A} \bullet \mathbf{A}))^\star \bullet \mathbf{A} \\
& + (\mathbf{T}_\bot + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \\
= \quad & \{\!\{ \text{ Denesting rule \& Equation (C.61) \& Distributivity of } \cdot \text{ on } + \}\!\} \\
& (\mathbf{A} \bullet \mathbf{T}_\bot)^\star \bullet \mathbf{A} \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}) \bullet \mathbf{A})^\star \bullet \mathbf{A} \\
& + (\mathbf{T}_\bot + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \\
= \quad & \{\!\{ \text{ Sliding rule \& Equation (C.60) } \}\!\} \\
& (\mathbf{A} \bullet \mathbf{T}_\bot)^\star \bullet (\mathbf{A} \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \bullet \mathbf{A} \\
& + (\mathbf{T}_\bot + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \\
= \quad & \{\!\{ \text{ Sliding rule \& Equation (C.61) \& Denesting rule } \}\!\} \\
& (\mathbf{T}_\bot + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}) \bullet \mathbf{A})^\star \\
& + (\mathbf{T}_\bot + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \\
= \quad & \{\!\{ \text{ Distributivity of } \cdot \text{ over } + \text{ \& Equations (C.63), (C.33) and (C.60) } \}\!\} \\
& (\mathbf{T}_\bot + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \\
& + (\mathbf{T}_\bot + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \\
= \quad & \{\!\{ \text{ Idempotency of } + \}\!\} \\
& (\mathbf{T}_\bot + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \ .
\end{aligned}
$$

For inequation (C.73),

$$
\begin{aligned}
& (\mathbf{T}_\bot + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \bullet \mathbf{WM} \\
= \quad & \{\!\{ \text{ Denesting rule \& Equations (C.63), (C.33), (C.60) and (C.61) } \}\!\} \\
& (\mathbf{A} \bullet \mathbf{T}_\bot)^\star \bullet \mathbf{A} \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} \bullet \mathbf{A} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A} \bullet \mathbf{A}))^\star \bullet \mathbf{WM} \\
= \quad & \{\!\{ \text{ Distributivity of } \cdot \text{ on } + \text{ \& Sliding rule } \}\!\} \\
& (\mathbf{A} \bullet \mathbf{T}_\bot)^\star \bullet (\mathbf{A} \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \bullet \mathbf{A} \bullet \mathbf{WM} \\
\leqslant \quad & \{\!\{ \text{ Inequation (C.59) \& Monotonicity of } \cdot \text{ and } + \text{ \& Equation (C.60) } \}\!\} \\
& (\mathbf{A} \bullet \mathbf{T}_\bot)^\star \bullet (\mathbf{A} \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \bullet \mathbf{A} \\
= \quad & \{\!\{ \text{ Sliding rule \& Equation (C.61) \& Denesting rule \& Distributivity of } \cdot \text{ on} \\
& \quad + \text{ \& Equations (C.63), (C.33) and (C.60) } \}\!\} \\
& (\mathbf{T}_\bot + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \ .
\end{aligned}
$$

For inequation (C.74),

$$
(\mathbf{T}_\bot + \mathbf{A})^\star \bullet ((\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{B} + \mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp}
$$

$$
\begin{aligned}
&= && \{\!\{ \text{ Definition of } {}^{+} \text{ } \}\!\} \\
&\quad (\mathbf{T}_{\perp} + \mathbf{A})^{\star} \bullet ((\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{B} + \mathbf{T}_{\mathsf{c}} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet \mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM} \bullet \varepsilon_{\not\perp} \\
&\leqslant && \{\!\{ \text{ Inequation (C.64) \& Monotonicity of } \cdot \text{ } \}\!\} \\
&\quad (\mathbf{T}_{\perp} + \mathbf{A})^{\star} \bullet ((\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{B} + \mathbf{T}_{\mathsf{c}} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet \mathbf{T}_{\mathsf{c}} \bullet (\mathbf{B} + \varepsilon_{\not\perp}) \\
&\leqslant && \{\!\{ \text{ Identity of } \cdot \text{ \& Fact: } \mathbf{I} \leqslant \mathbf{A} \text{ \& Monotonicity of } \cdot \text{ and } + \text{ } \}\!\} \\
&\quad (\mathbf{T}_{\perp} + \mathbf{A})^{\star} \bullet ((\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{B} + \mathbf{T}_{\mathsf{c}} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet \mathbf{T}_{\mathsf{c}} \bullet (\mathbf{B} + \varepsilon_{\not\perp} \bullet \mathbf{A}) \\
&= && \{\!\{ \text{ Distributivity of } \cdot \text{ over } + \text{ } \}\!\} \\
&\quad (\mathbf{T}_{\perp} + \mathbf{A})^{\star} \bullet ((\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{B} + \mathbf{T}_{\mathsf{c}} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{B} + \mathbf{T}_{\mathsf{c}} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}) \\
&\leqslant && \{\!\{ \text{ Kleene algebra: } p^{*} p \leqslant p^{*} \text{ \& Monotonicity of } \cdot \text{ } \}\!\} \\
&\quad (\mathbf{T}_{\perp} + \mathbf{A})^{\star} \bullet ((\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^{\star} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{B} + \mathbf{T}_{\mathsf{c}} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A}))^{\star} \, .
\end{aligned}
$$

### C.2.4 Proof of (C.59)

For inequation (C.59), first recall that a list of blocks $\mathcal{C}$ over labels $S \cup \{s_{\mathfrak{c}} \mid s \in S\}$ was used in the proof of Theorem C.1 such that a matrix $\mathbf{WM}_{\mathfrak{c}}$ defined by $\mathbf{WM}_{\mathfrak{c}}[s, s'] := (\!|_{s} \mathcal{C} |\!)^{s'}$ for $s, s' \in S$ is such that

$$\vdash \mathbf{WM} = \mathbf{I} + \mathbf{WM}_{\mathfrak{c}} \, . \tag{C.75}$$

Here, it is possible to do the same. So, it suffices to show that $\mathbf{I} + \mathbf{WM}_{\mathfrak{c}} \leqslant \mathbf{A}$. By a proof similar to Lemma 4.5, property i, $\mathbf{I} \leqslant \mathbf{A}$. It remains to prove that $\mathbf{WM}_{\mathfrak{c}} \leqslant \mathbf{A}$. This inequation follows easily from (3.37) using the substitution function $g_2 : S \cup \{s_{\mathfrak{c}} \mid s \in S\} \to \{s_{\mathfrak{a}_{\perp}}, s_{\mathfrak{a}_{\not\perp}}, s_{\mathfrak{p}_{\not\perp}} \mid s \in S\}$ defined by $g_2(s) := s_{\mathfrak{a}_{\perp}}$ and $g_2(s_{\mathfrak{c}}) := s_{\mathfrak{a}_{\not\perp}}$ for all $s \in S$, and noting that $\widehat{g_2}(\mathcal{C})$ is a sublist of $\mathcal{B}'$.

### C.2.5 Proofs of (C.60) and (C.61)

The proofs of (C.60) and (C.61) are similar to the proof of Lemma 4.5, properties ii and iii.

### C.2.6 Proofs of (C.62) and (C.63)

We first show that

$$\vdash \mathbf{WM} \bullet \mathbf{B} \bullet \mathbf{C} \leqslant \mathbf{B} \, . \tag{C.76}$$

For all $s, s' \in S$,

$$
\begin{array}{ll}
 & (\mathbf{WM} \bullet \mathbf{B} \bullet \mathbf{C})[s_{\mathfrak{a}\not\downarrow}, s'_{\mathfrak{p}\not\downarrow}] \\
= & \qquad \{\!\{ \text{ Definition of } \bullet \text{ } \}\!\} \\
 & (\sum t, t' \mid t, t' \in S : \mathbf{WM}[s, t] \cdot \mathbf{B}[t_{\mathfrak{a}\not\downarrow}, t'_{\mathfrak{p}\not\downarrow}] \cdot \mathbf{C}[t'_{\mathfrak{p}\not\downarrow}, s'_{\mathfrak{p}\not\downarrow}]) \\
= & \qquad \{\!\{ \text{ Definition of } \mathbf{WM}, \mathbf{B} \text{ and } \mathbf{C} \text{ \& Equation (C.32) } \}\!\} \\
 & (\sum t, t' \mid t, t' \in S : (\!|_{s_{\mathfrak{a}\not\downarrow}} \mathcal{B}' |\!)^{t_{\mathfrak{a}\not\downarrow}} \cdot (\!|_{t_{\mathfrak{a}\not\downarrow}} \mathcal{B}' |\!)^{t'_{\mathfrak{p}\not\downarrow}} \cdot (\!|_{t'_{\mathfrak{p}\not\downarrow}} \mathcal{B}' |\!)^{s'_{\mathfrak{p}\not\downarrow}}) \\
\leqslant & \qquad \{\!\{ \text{ Axiom (3.4) \& Monotonicity of } \cdot \text{ and } + \text{ } \}\!\} \\
 & (\sum t, t' \mid t, t' \in S : (\!|_{s_{\mathfrak{a}\not\downarrow}} \mathcal{B}' |\!)^{s'_{\mathfrak{p}\not\downarrow}}) \\
= & \qquad \{\!\{ \text{ Idempotency of } + \text{ } \}\!\} \\
 & (\!|_{s_{\mathfrak{a}\not\downarrow}} \mathcal{B}' |\!)^{s'_{\mathfrak{p}\not\downarrow}} \\
= & \qquad \{\!\{ \text{ Definition of } \mathbf{B} \text{ } \}\!\} \\
 & \mathbf{B}[s_{\mathfrak{a}\not\downarrow}, s'_{\mathfrak{p}\not\downarrow}] \ .
\end{array}
$$

We now prove (C.62). By Kleene algebra and Lemma 4.5, property i,

$$\mathbf{B} \quad = \quad \mathbf{I} \bullet \mathbf{B} \quad \leqslant \quad \mathbf{WM} \bullet \mathbf{B} \ .$$

Also, by Kleene algebra, the fact that $\mathbf{I} \leqslant \mathbf{A}$, and (C.33) and (C.76),

$$\mathbf{WM} \bullet \mathbf{B} \quad = \quad \mathbf{WM} \bullet \mathbf{B} \bullet \mathbf{I} \quad \leqslant \quad \mathbf{WM} \bullet \mathbf{B} \bullet \mathbf{A} \quad = \quad \mathbf{WM} \bullet \mathbf{B} \bullet \mathbf{C} \quad \leqslant \quad \mathbf{B} \ .$$

We now prove (C.63). By Kleene algebra, the fact that $\mathbf{I} \leqslant \mathbf{A}$, and (C.33),

$$\mathbf{B} \quad = \quad \mathbf{B} \bullet \mathbf{I} \quad \leqslant \quad \mathbf{B} \bullet \mathbf{A} \quad = \quad \mathbf{B} \bullet \mathbf{C} \ .$$

Also, by Kleene algebra, Lemma 4.5, property i, and (C.76),

$$\mathbf{B} \bullet \mathbf{C} \quad = \quad \mathbf{I} \bullet \mathbf{B} \bullet \mathbf{C} \quad \leqslant \quad \mathbf{WM} \bullet \mathbf{B} \bullet \mathbf{C} \quad \leqslant \quad \mathbf{B} \ .$$

### C.2.7 Proof of (C.64)

For inequation (C.64), we use the matrix $\mathbf{WM}_{\mathfrak{c}}$ like for the proof of inequation (C.59). So, using (C.75), distributivity of $\cdot$ over $+$ and identity of $\cdot$,

$$\mathbf{WM} \bullet \varepsilon_{\not\downarrow} \quad = \quad (\mathbf{I} + \mathbf{WM}_{\mathfrak{c}}) \bullet \varepsilon_{\not\downarrow} \quad = \quad \varepsilon_{\not\downarrow} + \mathbf{WM}_{\mathfrak{c}} \bullet \varepsilon_{\not\downarrow} \ .$$

Obviously, $\varepsilon_{\not\downarrow} \leqslant \varepsilon_{\not\downarrow}$. So, to prove (C.64), it suffices to show that, for all $s, s' \in S$,

$$\vdash (\mathbf{WM}_{\mathfrak{c}} \bullet \varepsilon_{\not\downarrow})[s, s'] \leqslant \mathbf{B}[s_{\mathfrak{a}\not\downarrow}, s'_{\mathfrak{p}\not\downarrow}] \ .$$

By the definition of $\bullet$ and $\varepsilon_{\not\perp}$ and Kleene algebra, it suffices to prove that, for all $s, s', s'' \in S$ such that $\vdash \varepsilon_{\not\perp}[s'', s'] = 1$,

$$\vdash \underset{s}{(\!|\, \mathcal{C} \,|\!)}^{s''} \leqslant \underset{s_{\mathfrak{a}_{\not\perp}}}{(\!|\, \mathcal{B}' \,|\!)}^{s'_{\mathfrak{p}_{\not\perp}}} \ .$$

Easily,

$$\begin{aligned}
& (\!|\, \mathcal{C} \,|\!)^{s''} \\
= \quad & \{\!\{ \text{ Definition of } \mathcal{C} \text{ \& Equation (3.25) } \}\!\} \\
& (\sum m \mid m \in \Sigma_{\mathbf{i}} \wedge [\!|\, m \,|\!]^{s''} \in \mathcal{C}^1 : m) \\
& + (\sum m, t \mid m \in \Sigma_{\mathbf{i}} \wedge [\!|\, t\, m \,|\!]^{s''} \in \mathcal{C}^1 : (\!|\, \mathcal{C} \,|\!)^t \cdot m) \\
& + (\sum c, z, r, w \mid [\!|\, c{\downarrow_{z_{\mathfrak{c}}}} {\uparrow^{w_{\mathfrak{c}}}}\, r \,|\!]^{s''} \in \mathcal{C}^2 : c \cdot (\!|\, \mathcal{C} \,|\!)^{w_{\mathfrak{c}}} \cdot r) \\
& + (\sum c, z, r, w, t \mid [\!|\, t\, c{\downarrow_{z_{\mathfrak{c}}}} {\uparrow^{w_{\mathfrak{c}}}}\, r \,|\!]^{s''} \in \mathcal{C}^2 : (\!|\, \mathcal{C} \,|\!)^t \cdot c \cdot (\!|\, \mathcal{C} \,|\!)^{w_{\mathfrak{c}}} \cdot r) \\
\leqslant \quad & \{\!\{ \text{ Hypothesis: } \vdash \varepsilon_{\not\perp}[s'', s'] = 1 \text{ \& Definition of } \mathcal{C} \text{ and } \mathcal{B}', \text{ conditions c and g:} \\
& \quad \text{if } m \in \Sigma_{\mathbf{i}} \wedge [\!|\, m \,|\!]^{s''} \in \mathcal{C}^1 \text{ and } \vdash \varepsilon_{\not\perp}[s'', s'] = 1, \text{ then } [\!|\, t_{\mathfrak{a}_{\not\perp}}\, m \,|\!]^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^1, \text{ and if} \\
& \quad [\!|\, t\, c{\downarrow_{z_{\mathfrak{c}}}} {\uparrow^{w_{\mathfrak{c}}}}\, r \,|\!]^{s''} \in \mathcal{C}^2 \text{ and} \vdash \varepsilon_{\not\perp}[s'', s'] = 1, \text{ then } [\!|\, t_{\mathfrak{a}_{\not\perp}}\, c{\downarrow_{z_{\mathfrak{a}_{\not\perp}}}} {\uparrow^{w_{\mathfrak{a}_{\not\perp}}}}\, r \,|\!]^{s''_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^2. \\
& \quad \text{\& Kleene algebra } \}\!\} \\
& (\sum m \mid [\!|\, s_{\mathfrak{a}_{\not\perp}}\, m \,|\!]^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^1 : m) \\
& + (\sum m, t \mid [\!|\, t_{\mathfrak{a}_{\not\perp}}\, m \,|\!]^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^1 : (\!|\, \mathcal{C} \,|\!)^t \cdot m) \\
& + (\sum c, z, r, w \mid [\!|\, s_{\mathfrak{a}_{\not\perp}}\, c{\downarrow_{z_{\mathfrak{a}_{\not\perp}}}} {\uparrow^{w_{\mathfrak{a}_{\not\perp}}}}\, r \,|\!]^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^2 : c \cdot (\!|\, \mathcal{C} \,|\!)^{w_{\mathfrak{c}}} \cdot r) \\
& + (\sum c, z, r, w, t \mid [\!|\, t_{\mathfrak{a}_{\not\perp}}\, c{\downarrow_{z_{\mathfrak{a}_{\not\perp}}}} {\uparrow^{w_{\mathfrak{a}_{\not\perp}}}}\, r \,|\!]^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^2 : (\!|\, \mathcal{C} \,|\!)^t \cdot c \cdot (\!|\, \mathcal{C} \,|\!)^{w_{\mathfrak{c}}} \cdot r) \\
\leqslant \quad & \{\!\{ \text{ Facts: } \mathbf{WM}_{\mathfrak{c}} \leqslant \mathbf{WM} \text{ and } (\!|\, z_{\mathfrak{c}}\, \mathcal{C} \,|\!)^{w_{\mathfrak{c}}} \leqslant (\!|\, z\, \mathcal{B} \,|\!)^{w} \text{ \& Equation (C.32): } (\!|\, u\, \mathcal{B} \,|\!)^{u'} = \\
& \quad (\!|\, u_{\mathfrak{a}_{\not\perp}}\, \mathcal{B}' \,|\!)^{u'_{\mathfrak{a}_{\not\perp}}} \text{ for } u, u' \in S \text{ \& Kleene algebra } \}\!\} \\
& (\sum m \mid [\!|\, s_{\mathfrak{a}_{\not\perp}}\, m \,|\!]^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^1 : m) \\
& + (\sum m, t \mid [\!|\, t_{\mathfrak{a}_{\not\perp}}\, m \,|\!]^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^1 : (\!|\, \mathcal{B}' \,|\!)^{t_{\mathfrak{a}_{\not\perp}}} \cdot m) \\
& + (\sum c, z, r, w \mid [\!|\, s_{\mathfrak{a}_{\not\perp}}\, c{\downarrow_{z_{\mathfrak{a}_{\not\perp}}}} {\uparrow^{w_{\mathfrak{a}_{\not\perp}}}}\, r \,|\!]^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^2 : c \cdot (\!|\, \mathcal{B}' \,|\!)^{w_{\mathfrak{a}_{\not\perp}}} \cdot r) \\
& + (\sum c, z, r, w, t \mid [\!|\, t_{\mathfrak{a}_{\not\perp}}\, c{\downarrow_{z_{\mathfrak{a}_{\not\perp}}}} {\uparrow^{w_{\mathfrak{a}_{\not\perp}}}}\, r \,|\!]^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^2 : (\!|\, \mathcal{B}' \,|\!)^{t_{\mathfrak{a}_{\not\perp}}} \cdot c \cdot (\!|\, \mathcal{B}' \,|\!)^{w_{\mathfrak{a}_{\not\perp}}} \cdot r) \\
\leqslant \quad & \{\!\{ \text{ Kleene algebra: } p_1 \leqslant p_1 + p_2 \text{ \& Equation (3.25) } \}\!\} \\
& (\!|\, s_{\mathfrak{a}_{\not\perp}}\, \mathcal{B}' \,|\!)^{s'_{\mathfrak{p}_{\not\perp}}} \ .
\end{aligned}$$

### C.2.8 Proofs of (C.65) and (C.66)

For inequations (C.65) and (C.66), we prove, for all $s, s' \in S$,

$$\vdash (\mathbf{T}_{\mathfrak{c}} \bullet \mathbf{B} \bullet \mathbf{T}_{\perp})[s, s'] \leqslant \mathbf{A}[s_{\mathfrak{a}_{\perp}}, s'_{\mathfrak{a}_{\perp}}] \ ,$$

$$\vdash (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{B} \bullet \mathbf{T}_{\perp})[s, s'] \leqslant \mathbf{B}[s_{\mathfrak{a}_{\not\perp}}, s'_{\mathfrak{p}_{\not\perp}}] \ .$$

Using the definition of $\bullet$, $\mathbf{A}$, $\mathbf{B}$, $\mathbf{T}_{\mathsf{c}}$ and $\mathbf{T}_{\perp}$ and Kleene algebra, it suffices to prove, for all $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $s, s', t, t' \in S$ such that $\vdash \mathbf{T}_{\mathsf{c}}[s, t] \geqslant c$ and $\vdash \mathbf{T}_{\perp}[t', s'] \geqslant r$,

$$\vdash c \cdot \underset{t_{\mathfrak{a}_{\not\perp}}}{(\!|} \mathcal{B}' \overset{t'_{\mathfrak{p}_{\not\perp}}}{|\!)} \cdot r \leqslant \underset{s_{\mathfrak{a}_{\perp}}}{(\!|} \mathcal{B}' \overset{s'_{\mathfrak{a}_{\perp}}}{|\!)} \ ,$$

$$\vdash c \cdot \underset{t_{\mathfrak{a}_{\not\perp}}}{(\!|} \mathcal{B}' \overset{t'_{\mathfrak{p}_{\not\perp}}}{|\!)} \cdot r \leqslant \underset{s_{\mathfrak{a}_{\not\perp}}}{(\!|} \mathcal{B}' \overset{s'_{\mathfrak{p}_{\not\perp}}}{|\!)} \ .$$

This is direct from the definition of $\mathcal{B}'$, conditions e and f, and axiom (3.3).

### C.2.9 Proofs of (C.67) and (C.68)

For inequations (C.67) and (C.68), we prove, for all $s, s' \in S$,

$$\vdash (\mathbf{T}_{\mathsf{c}} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A} \bullet \mathbf{T}_{\perp})[s, s'] \leqslant \mathbf{A}[s_{\mathfrak{a}_{\perp}}, s'_{\mathfrak{a}_{\perp}}] \ ,$$

$$\vdash (\mathbf{T}_{\mathsf{c}} \bullet \varepsilon_{\not\perp} \bullet \mathbf{A} \bullet \mathbf{T}_{\perp})[s, s'] \leqslant \mathbf{B}[s_{\mathfrak{a}_{\not\perp}}, s'_{\mathfrak{p}_{\not\perp}}] \ .$$

Using (C.33), the definition of $\bullet$, $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{T}_{\mathsf{c}}$, $\varepsilon_{\not\perp}$ and $\mathbf{T}_{\perp}$ and Kleene algebra, it suffices to prove, for all $c \in \Sigma_{\mathbf{c}}$, $r \in \Sigma_{\mathbf{r}}$ and $s, s', t, t', t'' \in S$ such that $\vdash \varepsilon_{\not\perp}[t'', t] = 1$, $\vdash \mathbf{T}_{\mathsf{c}}[s, t''] \geqslant c$ and $\vdash \mathbf{T}_{\perp}[t', s'] \geqslant r$,

$$\vdash c \cdot \underset{t_{\mathfrak{p}_{\not\perp}}}{(\!|} \mathcal{B}' \overset{t'_{\mathfrak{p}_{\not\perp}}}{|\!)} \cdot r \leqslant \underset{s_{\mathfrak{a}_{\perp}}}{(\!|} \mathcal{B}' \overset{s'_{\mathfrak{a}_{\perp}}}{|\!)} \ ,$$

$$\vdash c \cdot \underset{t_{\mathfrak{p}_{\not\perp}}}{(\!|} \mathcal{B}' \overset{t'_{\mathfrak{p}_{\not\perp}}}{|\!)} \cdot r \leqslant \underset{s_{\mathfrak{a}_{\not\perp}}}{(\!|} \mathcal{B}' \overset{s'_{\mathfrak{p}_{\not\perp}}}{|\!)} \ .$$

This is direct from the definition of $\mathcal{B}'$, conditions i and j, and axiom (3.3).

### C.2.10 Proofs of (C.69) and (C.70)

For inequations (C.69) and (C.70), we prove that

$$\vdash \mathbf{WM}' \leqslant \left[\begin{array}{c|c|c} (\mathbf{T}_\perp + \mathbf{WM} \\ \quad + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{WM} & (\mathbf{WM} + \mathbf{T_c})^\star \bullet \varepsilon_{\not\perp} \\ & & \bullet\ (\mathbf{T}_\perp + \mathbf{WM} \\ & & \quad + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star \\ \hline \mathbf{0} & \mathbf{0} & (\mathbf{T}_\perp + \mathbf{WM} \\ & & \quad + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star \end{array}\right] .$$

We use (3.5) with

$$\begin{aligned}
s_{(s_{\mathfrak{a}_\perp}, s'_{\mathfrak{a}_\perp})} &:= (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[s, s'] \ , \\
s_{(s_{\mathfrak{a}_\perp}, s'_{\mathfrak{p}_{\not\perp}})} &:= 0 \ , \\
s_{(s_{\mathfrak{a}_\perp}, s'_{\mathfrak{a}_{\not\perp}})} &:= 0 \ , \\
s_{(s_{\mathfrak{a}_{\not\perp}}, s'_{\mathfrak{a}_\perp})} &:= 0 \ , \\
s_{(s_{\mathfrak{a}_{\not\perp}}, s'_{\mathfrak{a}_{\not\perp}})} &:= (\!|_s \mathcal{B} |\!)^{s'} \ , \\
s_{(s_{\mathfrak{a}_{\not\perp}}, s'_{\mathfrak{p}_{\not\perp}})} &:= ((\mathbf{WM} + \mathbf{T_c})^\star \bullet \varepsilon_{\not\perp} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star)[s, s'] \ , \\
s_{(s_{\mathfrak{p}_{\not\perp}}, s'_{\mathfrak{a}_\perp})} &:= 0 \ , \\
s_{(s_{\mathfrak{p}_{\not\perp}}, s'_{\mathfrak{a}_{\not\perp}})} &:= 0 \ , \\
s_{(s_{\mathfrak{p}_{\not\perp}}, s'_{\mathfrak{p}_{\not\perp}})} &:= (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[s, s'] \ .
\end{aligned}$$

So, noting that every block of $\mathcal{B}'$ having as starting label a $t_{\mathfrak{a}_\perp}$ (respectively, $t_{\mathfrak{p}_{\not\perp}}$) must also have as ending label a $t'_{\mathfrak{a}_\perp}$ (respectively, $t'_{\mathfrak{p}_{\not\perp}}$) where $t, t' \in S$, and every block of $\mathcal{B}'$ having as starting label a $t_{\mathfrak{a}_{\not\perp}}$ must also have as ending label a $t'_{\mathfrak{a}_{\not\perp}}$ or $t'_{\mathfrak{p}_{\not\perp}}$ where $t, t' \in S$, it suffices to prove, for all $s, s' \in S$,

$$(\wedge\, m \mid \underset{s_{\mathfrak{a}_\perp}}{[\ m\ ]^{s'_{\mathfrak{a}_\perp}}} \in (\mathcal{B}')^1 : m \leqslant (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[s, s']) \ , \tag{C.77}$$

$$(\wedge\, m \mid \underset{s_{\mathfrak{a}_{\not\perp}}}{[\ m\ ]^{s'_{\mathfrak{a}_{\not\perp}}}} \in (\mathcal{B}')^1 : m \leqslant (\!|\underset{s}{\mathcal{B}}|\!)^{s'}) \ , \tag{C.78}$$

$$\begin{aligned}
&(\wedge\, m \mid \underset{s_{\mathfrak{a}_{\not\perp}}}{[\ m\ ]^{s'_{\mathfrak{p}_{\not\perp}}}} \in (\mathcal{B}')^1 : m \\
&\qquad \leqslant ((\mathbf{WM} + \mathbf{T_c})^\star \bullet \varepsilon_{\not\perp} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star)[s, s']) \ ,
\end{aligned} \tag{C.79}$$

$$(\wedge\, m \mid \underset{s_{\mathfrak{p}_{\not\perp}}}{[\ m\ ]^{s'_{\mathfrak{p}_{\not\perp}}}} \in (\mathcal{B}')^1 : m \leqslant (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[s, s']) \ , \tag{C.80}$$

$$(\wedge\, c, z, r, w \mid {\scriptstyle s_{\mathfrak{a}_\perp}}[\; c{\scriptstyle z}\downarrow\uparrow{\scriptstyle w} r \;]^{s'_{\mathfrak{a}_\perp}} \in (\mathcal{B}')^2 : c \cdot s_{(z,w)} \cdot r$$
$$\leqslant\ (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[s, s'])\ , \tag{C.81}$$

$$(\wedge\, c, z, r, w \mid {\scriptstyle s_{\mathfrak{a}_{\not\perp}}}[\; c{\scriptstyle z}\downarrow\uparrow{\scriptstyle w} r \;]^{s'_{\mathfrak{a}_{\not\perp}}} \in (\mathcal{B}')^2 : c \cdot s_{(z,w)} \cdot r \leqslant {\scriptstyle s}(\!|\, \mathcal{B} \,|\!)^{s'})\ , \tag{C.82}$$

$$(\wedge\, c, z, r, w \mid {\scriptstyle s_{\mathfrak{a}_{\not\perp}}}[\; c{\scriptstyle z}\downarrow\uparrow{\scriptstyle w} r \;]^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^2 : c \cdot s_{(z,w)} \cdot r$$
$$\leqslant\ ((\mathbf{WM} + \mathbf{T}_\mathsf{c})^\star \bullet \varepsilon_{\not\perp} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star)[s, s'])\ , \tag{C.83}$$

$$(\wedge\, c, z, r, w \mid {\scriptstyle s_{\mathfrak{p}_{\not\perp}}}[\; c{\scriptstyle z}\downarrow\uparrow{\scriptstyle w} r \;]^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^2 : c \cdot s_{(z,w)} \cdot r$$
$$\leqslant\ (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[s, s'])\ , \tag{C.84}$$

$$(\wedge\, v \mid v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\} : s_{(s_{\mathfrak{a}_\perp}, v)} \cdot s_{(v, s'_{\mathfrak{a}_\perp})}$$
$$\leqslant\ (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[s, s'])\ , \tag{C.85}$$

$$(\wedge\, v \mid v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\} : s_{(s_{\mathfrak{a}_\perp}, v)} \cdot s_{(v, s'_{\mathfrak{a}_{\not\perp}})} \leqslant 0)\ , \tag{C.86}$$

$$(\wedge\, v \mid v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\} : s_{(s_{\mathfrak{a}_\perp}, v)} \cdot s_{(v, s'_{\mathfrak{p}_{\not\perp}})} \leqslant 0)\ , \tag{C.87}$$

$$(\wedge\, v \mid v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\} : s_{(s_{\mathfrak{a}_{\not\perp}}, v)} \cdot s_{(v, s'_{\mathfrak{a}_\perp})} \leqslant 0)\ , \tag{C.88}$$

$$(\wedge\, v \mid v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\} : s_{(s_{\mathfrak{a}_{\not\perp}}, v)} \cdot s_{(v, s'_{\mathfrak{a}_{\not\perp}})} \leqslant {\scriptstyle s}(\!|\, \mathcal{B} \,|\!)^{s'})\ , \tag{C.89}$$

$$(\wedge\, v \mid v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\} : s_{(s_{\mathfrak{a}_{\not\perp}}, v)} \cdot s_{(v, s'_{\mathfrak{p}_{\not\perp}})}$$
$$\leqslant\ ((\mathbf{WM} + \mathbf{T}_\mathsf{c})^\star \bullet \varepsilon_{\not\perp} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star)[s, s'])\ , \tag{C.90}$$

$$(\wedge\, v \mid v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\} : s_{(s_{\mathfrak{p}_{\not\perp}}, v)} \cdot s_{(v, s'_{\mathfrak{a}_\perp})} \leqslant 0)\ , \tag{C.91}$$

$$(\wedge\, v \mid v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\} : s_{(s_{\mathfrak{p}_{\not\perp}}, v)} \cdot s_{(v, s'_{\mathfrak{a}_{\not\perp}})} \leqslant 0)\ , \tag{C.92}$$

$$(\wedge\, v \mid v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\} : s_{(s_{\mathfrak{p}_{\not\perp}}, v)} \cdot s_{(v, s'_{\mathfrak{p}_{\not\perp}})}$$
$$\leqslant\ (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[s, s'])\ . \tag{C.93}$$

For inequations (C.77), we suppose ${\scriptstyle s_{\mathfrak{a}_\perp}}[\, m \,]^{s'_{\mathfrak{a}_\perp}} \in (\mathcal{B}')^1$ and we prove

$$m \leqslant (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[s, s']\ .$$

By Kleene algebra, it suffices to prove

$$m \leqslant \mathbf{WM}[s, s']\ .$$

By the definition of $\mathbf{WM}$, it suffices to prove

$$m \leqslant {\scriptstyle s}(\!|\, \mathcal{B} \,|\!)^{s'}\ .$$

By the definition of $\mathcal{B}'$, conditions a and b, and the definition of $\mathcal{B}$, $[\!_s\, m\,]^{s'} \in \mathcal{B}^1$. So, the proof follows from axiom (3.2).

For inequations (C.78), we suppose $[\!_{s_{\mathfrak{a}_\not\perp}}\, m\,]^{s'_{\mathfrak{a}_\not\perp}} \in (\mathcal{B}')^1$ and we prove

$$m \leqslant (\!|\, \mathcal{B} \,|\!)^{s'}_{s} \quad .$$

By the definition of $\mathcal{B}'$, conditions a and b, and the definition of $\mathcal{B}$, $[\!_s\, m\,]^{s'} \in \mathcal{B}^1$. So, the proof follows from axiom (3.2).

For inequations (C.79), we suppose $[\!_{s_{\mathfrak{a}_\not\perp}}\, m\,]^{s'_{\mathfrak{p}_\not\perp}} \in (\mathcal{B}')^1$ and we prove

$$m \leqslant ((\mathbf{WM} + \mathbf{T}_{\mathsf{c}})^\star \bullet \varepsilon_\not\perp \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star)[s, s'] \quad .$$

By Kleene algebra, it suffices to prove

$$m \leqslant (\mathbf{WM} \bullet \varepsilon_\not\perp)[s, s'] \quad .$$

By the definition of $\bullet$, it suffices to prove

$$m \leqslant (\sum t \mid t \in S : \mathbf{WM}[s, t] \cdot \varepsilon_\not\perp[t, s']) \quad .$$

By the definition of $\mathbf{WM}$, it suffices to prove

$$m \leqslant (\sum t \mid t \in S : (\!|\, \mathcal{B} \,|\!)^{t}_{s} \cdot \varepsilon_\not\perp[t, s']) \quad .$$

We now use the hypothesis $[\!_{s_{\mathfrak{a}_\not\perp}}\, m\,]^{s'_{\mathfrak{p}_\not\perp}} \in (\mathcal{B}')^1$. By the definition of $\mathcal{B}'$, condition c, there exists a $s'' \in S$ such that $\vdash \varepsilon_\not\perp[s'', s'] = 1$ and $[\!_s\, m\,]^{s''} \in \mathcal{B}^1$. So, using axiom (3.2), Kleene algebra and hypothesis $\vdash \varepsilon_\not\perp[s'', s'] = 1$, it suffices to prove

$$(\!|\, \mathcal{B} \,|\!)^{s''}_{s} \cdot \varepsilon_\not\perp[s'', s'] \leqslant (\sum t \mid t \in S : (\!|\, \mathcal{B} \,|\!)^{t}_{s} \cdot \varepsilon_\not\perp[t, s']) \quad .$$

This is direct by Kleene algebra ($p_1 \leqslant p_1 + p_2$).

The proofs of inequations (C.80) are similar to the proofs of (C.77).

For inequations (C.81), we suppose $[\!_{s_{\mathfrak{a}_\perp}}\, c{\downarrow_z}{\uparrow^w}\, r\,]^{s'_{\mathfrak{a}_\perp}} \in (\mathcal{B}')^2$ and we prove

$$c \cdot s_{(z,w)} \cdot r \leqslant (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star[s, s'] \quad . \tag{C.94}$$

By the definition of $\mathcal{B}'$, conditions d, e, h and i, $z \in \{z'_{\mathfrak{a}_\not\perp}, z'_{\mathfrak{p}_\not\perp} \mid z' \in S\}$ and $w \in \{w'_{\mathfrak{a}_\not\perp}, w'_{\mathfrak{p}_\not\perp} \mid w' \in S\}$.

If $z = z'_{\mathfrak{a}_\not\varepsilon}$ for a $z' \in S$ and $w = w'_{\mathfrak{a}_\not\varepsilon}$ for a $w' \in S$, then, by condition d of the definition of $\mathcal{B}'$, $[\!_s\, c{\downarrow_{z'}} {\uparrow^{w'}}\, r\,]^{s'} \in \mathcal{B}^2$. Also,

$$s_{(z'_{\mathfrak{a}_\not\varepsilon}, w'_{\mathfrak{a}_\not\varepsilon})} = \overset{w'}{\underset{z'}{(\!|\, \mathcal{B}\, |\!)}} \quad .$$

So, using axiom (3.3) and the definition of $\mathbf{WM}$,

$$c \cdot s_{(z'_{\mathfrak{a}_\not\varepsilon}, w'_{\mathfrak{a}_\not\varepsilon})} \cdot r \quad = \quad c \cdot \overset{w'}{\underset{z'}{(\!|\, \mathcal{B}\, |\!)}} \cdot r \quad \leqslant \quad \overset{s'}{\underset{s}{(\!|\, \mathcal{B}\, |\!)}} \quad = \quad \mathbf{WM}[s, s'] \ .$$

Then, inequation (C.94) is trivial by Kleene algebra ($p_1 \leqslant (p_1 + p_2)^*$).

If $z = z'_{\mathfrak{a}_\not\varepsilon}$ for a $z' \in S$ and $w = w'_{\mathfrak{p}_\not\varepsilon}$ for a $w' \in S$, then, by condition e of the definition of $\mathcal{B}'$, $\vdash \mathbf{T}_\mathsf{c}[s, z'] \geqslant c$ and $\vdash \mathbf{T}_\perp[w', s'] \geqslant r$. Also,

$$s_{(z'_{\mathfrak{a}_\not\varepsilon}, w'_{\mathfrak{p}_\not\varepsilon})} = ((\mathbf{WM} + \mathbf{T}_\mathsf{c})^\star \bullet \varepsilon_\not\varepsilon \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\varepsilon)^\star)[z', w'] \ .$$

Then, inequation (C.94) is possible to prove.

$$\begin{array}{ll}
 & c \cdot s_{(z'_{\mathfrak{a}_\not\varepsilon}, w'_{\mathfrak{p}_\not\varepsilon})} \cdot r \\
= & \quad \{\!\!\{ \text{ Definition of } s_{(z'_{\mathfrak{a}_\not\varepsilon}, w'_{\mathfrak{p}_\not\varepsilon})} \}\!\!\} \\
 & c \cdot ((\mathbf{WM} + \mathbf{T}_\mathsf{c})^\star \bullet \varepsilon_\not\varepsilon \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\varepsilon)^\star)[z', w'] \cdot r \\
\leqslant & \quad \{\!\!\{ \text{ Hypotheses: } \vdash \mathbf{T}_\mathsf{c}[s, z'] \geqslant c \text{ and } \vdash \mathbf{T}_\perp[w', s'] \geqslant r \ \}\!\!\} \\
 & \mathbf{T}_\mathsf{c}[s, z'] \cdot ((\mathbf{WM} + \mathbf{T}_\mathsf{c})^\star \bullet \varepsilon_\not\varepsilon \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\varepsilon)^\star)[z', w'] \cdot \mathbf{T}_\perp[w', s'] \\
\leqslant & \quad \{\!\!\{ \text{ Kleene algebra \& Definition of } \bullet \ \}\!\!\} \\
 & (\mathbf{T}_\mathsf{c} \bullet (\mathbf{WM} + \mathbf{T}_\mathsf{c})^\star \bullet \varepsilon_\not\varepsilon \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\varepsilon)^\star \bullet \mathbf{T}_\perp)[s, s'] \\
= & \quad \{\!\!\{ \text{ Denesting rule \& Lemma 4.5, property iii } \}\!\!\} \\
 & (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM} \bullet (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^\star \bullet \varepsilon_\not\varepsilon \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\varepsilon)^\star \bullet \mathbf{T}_\perp)[s, s'] \\
= & \quad \{\!\!\{ \text{ Definition of } {}^+ \ \}\!\!\} \\
 & ((\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\varepsilon \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\varepsilon)^\star \bullet \mathbf{T}_\perp)[s, s'] \\
\leqslant & \quad \{\!\!\{ \text{ Kleene algebra: } p_1 \leqslant (p_1 + p_2)^* \text{ \& Monotonicity of } \cdot \text{ \& Kleene algebra: } \\
 & \qquad p^* = p^* p^* \ \}\!\!\} \\
 & (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\varepsilon)^\star[s, s']
\end{array}$$

If $z = z'_{\mathfrak{p}_\not\varepsilon}$ for a $z' \in S$ and $w = w'_{\mathfrak{a}_\not\varepsilon}$ for a $w' \in S$, then $s_{(z'_{\mathfrak{p}_\not\varepsilon}, w'_{\mathfrak{a}_\not\varepsilon})} = 0$. So, the proof of (C.94) follows from zero of $\cdot$ and the fact that 0 is the minimum of the algebra.

If $z = z'_{\mathfrak{p}_\not\varepsilon}$ for a $z' \in S$ and $w = w'_{\mathfrak{p}_\not\varepsilon}$ for a $w' \in S$, then, by condition i of the definition of $\mathcal{B}'$, there exists a $t \in S$ such that $\vdash \varepsilon_\not\varepsilon[t, z'] = 1$, $\vdash \mathbf{T}_\mathsf{c}[s, t] \geqslant c$ and $\vdash \mathbf{T}_\perp[w', s'] \geqslant r$. Also,

$$s_{(z'_{\mathfrak{p}_\not\varepsilon}, w'_{\mathfrak{p}_\not\varepsilon})} = (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\varepsilon)^\star[z', w'] \ .$$

Then, inequation (C.94) is possible to prove.

$$
\begin{array}{ll}
 & c \cdot s_{(z'_{\mathfrak{p}_\not\perp}, w'_{\mathfrak{p}_\not\perp})} \cdot r \\
= & \quad \{\!\!\{ \text{ Definition of } s_{(z'_{\mathfrak{p}_\not\perp}, w'_{\mathfrak{p}_\not\perp})} \}\!\!\} \\
 & c \cdot (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star[z', w'] \cdot r \\
= & \quad \{\!\!\{ \text{ Identity of } \cdot \}\!\!\} \\
 & c \cdot 1 \cdot 1 \cdot (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star[z', w'] \cdot r \\
\leqslant & \quad \{\!\!\{ \text{ Hypotheses: } \vdash \varepsilon_\not\perp[t, z'] = 1, \vdash \mathbf{T}_\mathsf{c}[s, t] \geqslant c \text{ and } \vdash \mathbf{T}_\perp[w', s'] \geqslant r \ \& \\
 & \quad \text{Lemma 4.5, property i \& Definition of } \mathbf{I} \}\!\!\} \\
 & \mathbf{T}_\mathsf{c}[s, t] \cdot \mathbf{WM}[t, t] \cdot \varepsilon_\not\perp[t, z'] \cdot (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star[z', w'] \cdot \mathbf{T}_\perp[w', s'] \\
\leqslant & \quad \{\!\!\{ \text{ Kleene algebra \& Definition of } \bullet \}\!\!\} \\
 & (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM} \bullet \varepsilon_\not\perp \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star \bullet \mathbf{T}_\perp)[s, s'] \\
\leqslant & \quad \{\!\!\{ \text{ Kleene algebra: } p \leqslant p^+ \text{ \& Monotonicity of } \cdot \}\!\!\} \\
 & ((\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star \bullet \mathbf{T}_\perp)[s, s'] \\
\leqslant & \quad \{\!\!\{ \text{ Kleene algebra: } p_1 \leqslant (p_1 + p_2)^* \text{ \& Monotonicity of } \cdot \text{ \& Kleene algebra:} \\
 & \quad p^* = p^* p^* \}\!\!\} \\
 & (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star[s, s']
\end{array}
$$

For inequations (C.82), we suppose $\lfloor_{s_{\mathfrak{a}_\not\perp}} c \downarrow_z \uparrow^w r \rceil^{s'_{\mathfrak{a}_\not\perp}} \in (\mathcal{B}')^2$ and we prove

$$c \cdot s_{(z,w)} \cdot r \leqslant \underset{s}{\overset{s'}{(\!|\, \mathcal{B} \,|\!)}} \quad . \tag{C.95}$$

By the definition of $\mathcal{B}'$, conditions d and h, $z \in \{z'_{\mathfrak{a}_\not\perp}, z'_{\mathfrak{p}_\not\perp} \mid z' \in S\}$ and $w \in \{w'_{\mathfrak{a}_\not\perp} \mid w' \in S\}$. If $z = z'_{\mathfrak{a}_\not\perp}$ for a $z' \in S$ and $w = w'_{\mathfrak{a}_\not\perp}$ for a $w' \in S$, then $s_{(z'_{\mathfrak{a}_\not\perp}, w'_{\mathfrak{a}_\not\perp})} = \underset{z'}{(\!|\, \mathcal{B} \,|\!)^{w'}}$ and $\lfloor_s c \downarrow_{z'} \uparrow^{w'} r \rceil^{s'} \in \mathcal{B}^2$. So, the proof of (C.95) follows from (3.3). If $z = z'_{\mathfrak{p}_\not\perp}$ for a $z' \in S$ and $w = w'_{\mathfrak{a}_\not\perp}$ for a $w' \in S$, then $s_{(z'_{\mathfrak{p}_\not\perp}, w'_{\mathfrak{a}_\not\perp})} = 0$. The proof of (C.95) follows from zero of $\cdot$ and the fact that 0 is the minimum of the algebra.

For inequations (C.83), we suppose $\lfloor_{s_{\mathfrak{a}_\not\perp}} c \downarrow_z \uparrow^w r \rceil^{s'_{\mathfrak{p}_\not\perp}} \in (\mathcal{B}')^2$ and we prove

$$c \cdot s_{(z,w)} \cdot r \leqslant ((\mathbf{WM} + \mathbf{T}_\mathsf{c})^\star \bullet \varepsilon_\not\perp \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_\mathsf{c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star)[s, s'] \quad . \tag{C.96}$$

By the definition of $\mathcal{B}'$, conditions f, g, j and k, $z \in \{z'_{\mathfrak{a}_\not\perp}, z'_{\mathfrak{p}_\not\perp} \mid z' \in S\}$ and $w \in \{w'_{\mathfrak{a}_\not\perp}, w'_{\mathfrak{p}_\not\perp} \mid w' \in S\}$.

If $z = z'_{\mathfrak{a}_\not\perp}$ for a $z' \in S$ and $w = w'_{\mathfrak{a}_\not\perp}$ for a $w' \in S$, then, by condition g of the definition of $\mathcal{B}'$, there exists a $t \in S$ such that $\vdash \varepsilon_\not\perp[t, s'] = 1$ and $\lfloor_s c \downarrow_{z'} \uparrow^{w'} r \rceil^t \in \mathcal{B}^2$.

Also,

$$s_{(z'_{\mathtt{a}_{\not\perp}},w'_{\mathtt{a}_{\not\perp}})} = \big(\!\!\big|\, \mathcal{B} \,\big|\!\!\big)^{w'}_{z'} \quad .$$

So, inequation (C.96) is possible to prove.

$$\begin{array}{ll}
& c \cdot s_{(z'_{\mathtt{a}_{\not\perp}},w'_{\mathtt{a}_{\not\perp}})} \cdot r \\
= & \qquad \{\!\!\{ \text{ Definition of } s_{(z'_{\mathtt{a}_{\not\perp}},w'_{\mathtt{a}_{\not\perp}})} \,\}\!\!\} \\
& c \cdot \big(\!\!\big|_{z'} \mathcal{B} \,\big|\!\!\big)^{w'} \cdot r \\
\leqslant & \qquad \{\!\!\{ \text{ Hypothesis: } [_s\, c{\downarrow_{z'}} {\uparrow^{w'}}\, r\,]^t \in \mathcal{B}^2 \text{ \& Axiom (3.3)} \,\}\!\!\} \\
& \big(\!\!\big|_s \mathcal{B} \,\big|\!\!\big)^t \\
= & \qquad \{\!\!\{ \text{ Definition of } \mathbf{WM} \text{ \& Identity of } \cdot \,\}\!\!\} \\
& \mathbf{WM}[s,t] \cdot 1 \\
= & \qquad \{\!\!\{ \text{ Hypothesis: } \vdash \varepsilon_{\not\perp}[t,s'] = 1 \,\}\!\!\} \\
& \mathbf{WM}[s,t] \cdot \varepsilon_{\not\perp}[t,s'] \\
\leqslant & \qquad \{\!\!\{ \text{ Kleene algebra \& Definition of } \bullet \,\}\!\!\} \\
& (\mathbf{WM} \bullet \varepsilon_{\not\perp})[s,s'] \\
\leqslant & \qquad \{\!\!\{ \text{ Kleene algebra: } p_1 \leqslant (p_1 + p_2)^* \text{ \& Monotonicity of } \cdot \text{ \& Identity of } \cdot \,\}\!\!\} \\
& ((\mathbf{WM} + \mathbf{T}_{\mathtt{c}})^\star \bullet \varepsilon_{\not\perp} \bullet \mathbf{I})[s,s'] \\
\leqslant & \qquad \{\!\!\{ \text{ Kleene algebra: } 1 \leqslant p^* \text{ \& Monotonicity of } \cdot \,\}\!\!\} \\
& ((\mathbf{WM} + \mathbf{T}_{\mathtt{c}})^\star \bullet \varepsilon_{\not\perp} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_{\mathtt{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star)[s,s']
\end{array}$$

If $z = z'_{\mathtt{a}_{\not\perp}}$ for a $z' \in S$ and $w = w'_{\mathtt{p}_{\not\perp}}$ for a $w' \in S$, then, by condition f of the definition of $\mathcal{B}'$, $\vdash \mathbf{T}_{\mathtt{c}}[s,z'] \geqslant c$ and $\vdash \mathbf{T}_\perp[w',s'] \geqslant r$. Also,

$$s_{(z'_{\mathtt{a}_{\not\perp}},w'_{\mathtt{p}_{\not\perp}})} = ((\mathbf{WM} + \mathbf{T}_{\mathtt{c}})^\star \bullet \varepsilon_{\not\perp} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_{\mathtt{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star)[z',w'] \quad .$$

Then, inequation (C.96) is possible to prove.

$$\begin{array}{ll}
& c \cdot s_{(z'_{\mathtt{a}_{\not\perp}},w'_{\mathtt{p}_{\not\perp}})} \cdot r \\
= & \qquad \{\!\!\{ \text{ Definition of } s_{(z'_{\mathtt{a}_{\not\perp}},w'_{\mathtt{p}_{\not\perp}})} \,\}\!\!\} \\
& c \cdot ((\mathbf{WM} + \mathbf{T}_{\mathtt{c}})^\star \bullet \varepsilon_{\not\perp} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_{\mathtt{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star)[z',w'] \cdot r \\
\leqslant & \qquad \{\!\!\{ \text{ Hypotheses: } \vdash \mathbf{T}_{\mathtt{c}}[s,z'] \geqslant c \text{ and } \vdash \mathbf{T}_\perp[w',s'] \geqslant r \,\}\!\!\} \\
& \mathbf{T}_{\mathtt{c}}[s,z'] \cdot ((\mathbf{WM} + \mathbf{T}_{\mathtt{c}})^\star \bullet \varepsilon_{\not\perp} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_{\mathtt{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star)[z',w'] \cdot \mathbf{T}_\perp[w',s'] \\
\leqslant & \qquad \{\!\!\{ \text{ Kleene algebra \& Definition of } \bullet \,\}\!\!\} \\
& (\mathbf{T}_{\mathtt{c}} \bullet (\mathbf{WM} + \mathbf{T}_{\mathtt{c}})^\star \bullet \varepsilon_{\not\perp} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T}_{\mathtt{c}} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star \bullet \mathbf{T}_\perp)[s,s'] \\
\leqslant & \qquad \{\!\!\{ \text{ Kleene algebra: } p_1 \leqslant (p_1 + p_2)^* \text{ \& Monotonicity of } \cdot \text{ \& Kleene algebra:} \\
& \qquad\quad p^* = p^* p^* \,\}\!\!\}
\end{array}$$

$$
\begin{array}{ll}
 & ((\mathbf{WM}+\mathbf{T_c})^\star \bullet \varepsilon_{\not\perp} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star \bullet \mathbf{T}_\perp)[s,s'] \\
\leqslant & \qquad \{\!\{ \text{Kleene algebra: } p_1 \leqslant (p_1+p_2)^* \text{ \& Monotonicity of } \cdot \text{ \& Kleene algebra: } \\
 & \qquad\quad p^* = p^*p^* \,\}\!\} \\
 & ((\mathbf{WM}+\mathbf{T_c})^\star \bullet \varepsilon_{\not\perp} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star)[s,s']
\end{array}
$$

If $z = z'_{\mathfrak{p}_{\not\perp}}$ for a $z' \in S$ and $w = w'_{\mathfrak{a}_{\not\perp}}$ for a $w' \in S$, then $s_{(z'_{\mathfrak{p}_{\not\perp}}, w'_{\mathfrak{a}_{\not\perp}})} = 0$. So, the proof of (C.96) follows from zero of $\cdot$ and the fact that 0 is the minimum of the algebra.

If $z = z'_{\mathfrak{p}_{\not\perp}}$ for a $z' \in S$ and $w = w'_{\mathfrak{p}_{\not\perp}}$ for a $w' \in S$, then, by condition j of the definition of $\mathcal{B}'$, there exists a $t \in S$ such that $\vdash \varepsilon_{\not\perp}[t,z'] = 1$, $\vdash \mathbf{T_c}[s,t] \geqslant c$ and $\vdash \mathbf{T}_\perp[w',s'] \geqslant r$. Also,

$$s_{(z'_{\mathfrak{p}_{\not\perp}}, w'_{\mathfrak{p}_{\not\perp}})} = (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[z',w'] \ .$$

Then, inequation (C.96) is possible to prove.

$$
\begin{array}{ll}
 & c \cdot s_{(z'_{\mathfrak{p}_{\not\perp}}, w'_{\mathfrak{p}_{\not\perp}})} \cdot r \\
= & \qquad \{\!\{ \text{Definition of } s_{(z'_{\mathfrak{p}_{\not\perp}}, w'_{\mathfrak{p}_{\not\perp}})} \,\}\!\} \\
 & c \cdot (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[z',w'] \cdot r \\
= & \qquad \{\!\{ \text{Identity of } \cdot \,\}\!\} \\
 & c \cdot 1 \cdot (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[z',w'] \cdot r \\
\leqslant & \qquad \{\!\{ \text{Hypotheses: } \vdash \varepsilon_{\not\perp}[t,z'] = 1, \vdash \mathbf{T_c}[s,t] \geqslant c \text{ and } \vdash \mathbf{T}_\perp[w',s'] \geqslant r \,\}\!\} \\
 & \mathbf{T_c}[s,t] \cdot \varepsilon_{\not\perp}[t,z'] \cdot (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[z',w'] \cdot \mathbf{T}_\perp[w',s'] \\
\leqslant & \qquad \{\!\{ \text{Kleene algebra \& Definition of } \bullet \,\}\!\} \\
 & (\mathbf{T_c} \bullet \varepsilon_{\not\perp} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star \bullet \mathbf{T}_\perp)[s,s'] \\
\leqslant & \qquad \{\!\{ \text{Kleene algebra: } p_1 \leqslant (p_1+p_2)^* \text{ \& Monotonicity of } \cdot \,\}\!\} \\
 & ((\mathbf{WM}+\mathbf{T_c})^\star \bullet \varepsilon_{\not\perp} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star \bullet \mathbf{T}_\perp)[s,s'] \\
\leqslant & \qquad \{\!\{ \text{Kleene algebra: } p_1 \leqslant (p_1+p_2)^* \text{ \& Monotonicity of } \cdot \text{ \& Kleene algebra: } \\
 & \qquad\quad p^* = p^*p^* \,\}\!\} \\
 & ((\mathbf{WM}+\mathbf{T_c})^\star \bullet \varepsilon_{\not\perp} \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star)[s,s']
\end{array}
$$

For inequations (C.84), we suppose $[_{s_{\mathfrak{p}_{\not\perp}}} c{\downarrow_z}{\uparrow^w} r \,]^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^2$ and we prove

$$c \cdot s_{(z,w)} \cdot r \leqslant (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[s,s'] \ .$$

By the definition of $\mathcal{B}'$, conditions d, e, f, h, i and j,

$$[_{s_{\mathfrak{p}_{\not\perp}}} c{\downarrow_z}{\uparrow^w} r \,]^{s'_{\mathfrak{p}_{\not\perp}}} \in (\mathcal{B}')^2 \Leftrightarrow [_{s_{\mathfrak{a}_\perp}} c{\downarrow_z}{\uparrow^w} r \,]^{s'_{\mathfrak{a}_\perp}} \in (\mathcal{B}')^2 \ .$$

So, the proof follows from (C.81).

For inequations (C.85), there are two possible cases for $v$. If $v = v'_{\mathfrak{a}_\perp}$ for a $v' \in S$, then it must be proved that

$$(\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[s, v'] \cdot (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[v', s'] \leqslant (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[s, s'] \ .$$

By Kleene algebra and the definition of $\bullet$, it suffices to prove that

$$\Big((\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star\Big)[s, s'] \leqslant (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[s, s'] \ .$$

This is trivial since $p^* = p^*p^*$ by Kleene algebra. If $v \in \{v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\}$, then it must be proved that

$$0 \cdot s_{(v, s'_{\mathfrak{a}_\perp})} \leqslant (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[s, s'] \ .$$

This is direct by zero of $\cdot$ and the fact that 0 is the minimum element of the algebra.

For inequations (C.86), there are two possible cases for $v$. If $v = v'_{\mathfrak{a}_\perp}$ for a $v' \in S$, then it must be proved that

$$(\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[s, v'] \cdot 0 \leqslant 0 \ .$$

This is direct by zero of $\cdot$ and the reflexivity of $\leqslant$. If $v \in \{v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\}$, then it must be proved that

$$0 \cdot s_{(v, s'_{\mathfrak{a}_{\not\perp}})} \leqslant 0 \ .$$

This is direct by zero of $\cdot$ and the reflexivity of $\leqslant$.

The proofs of (C.87) are similar to the proofs of (C.86).

For inequations (C.88), there are two possible cases for $v$. If $v = v'_{\mathfrak{a}_\perp}$ for a $v' \in S$, then it must be proved that

$$0 \cdot (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_{\not\perp})^\star[v', s'] \leqslant 0 \ .$$

This is direct by zero of $\cdot$ and the reflexivity of $\leqslant$. If $v \in \{v'_{\mathfrak{a}_{\not\perp}}, v'_{\mathfrak{p}_{\not\perp}} \mid v' \in S\}$, then it must be proved that

$$s_{(s_{\mathfrak{a}_{\not\perp}}, v)} \cdot 0 \leqslant 0 \ .$$

This is direct by zero of $\cdot$ and the reflexivity of $\leqslant$.

For inequations (C.89), there are two possible cases for $v$. If $v = v'_{\mathfrak{a}_\not\perp}$ for a $v' \in S$, then it must be proved that

$$\left(\!\!\left|\, \mathcal{B} \,\right|\!\!\right)^{v'}_{s} \cdot \left(\!\!\left|\, \mathcal{B} \,\right|\!\!\right)^{s'}_{v'} \leqslant \left(\!\!\left|\, \mathcal{B} \,\right|\!\!\right)^{s'}_{s} \quad .$$

This is direct by axiom (3.4). If $v \in \{v'_{\mathfrak{a}_\perp}, v'_{\mathfrak{p}_\not\perp} \mid v' \in S\}$, then it must be proved that

$$s_{(s_{\mathfrak{a}_\not\perp}, v)} \cdot 0 \leqslant \left(\!\!\left|\, \mathcal{B} \,\right|\!\!\right)^{s'}_{s} \quad .$$

This is direct by zero of $\cdot$ and the fact that 0 is the minimum element of the algebra.

For inequations (C.90), there are three possible cases for $v$. If $v = v'_{\mathfrak{a}_\perp}$ for a $v' \in S$, then it must be proved that

$$0 \cdot 0 \leqslant ((\mathbf{WM} + \mathbf{T_c})^\star \bullet \varepsilon_\not\perp \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star)[s, s'] \quad .$$

This is direct by zero of $\cdot$ and the fact that 0 is the minimum element of the algebra. If $v = v'_{\mathfrak{a}_\not\perp}$ for a $v' \in S$, then it must be proved that

$$\begin{aligned} \left(\!\!\left|\, \mathcal{B} \,\right|\!\!\right)^{v'}_{s} \cdot ((\mathbf{WM} + \mathbf{T_c})^\star \bullet \varepsilon_\not\perp \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star)[v', s'] \\ \leqslant ((\mathbf{WM} + \mathbf{T_c})^\star \bullet \varepsilon_\not\perp \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star)[s, s'] \quad . \end{aligned}$$

By the definition of $\mathbf{WM}$, by Kleene algebra and by the definition of $\bullet$, it suffices to prove that

$$\begin{aligned} (\mathbf{WM} \bullet (\mathbf{WM} + \mathbf{T_c})^\star \bullet \varepsilon_\not\perp \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star)[s, s'] \\ \leqslant ((\mathbf{WM} + \mathbf{T_c})^\star \bullet \varepsilon_\not\perp \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star)[s, s'] \quad . \end{aligned}$$

This is trivial since $p_1 \cdot (p_1 + p_2)^* \leqslant (p_1 + p_2)^*$ by Kleene algebra. If $v = v'_{\mathfrak{p}_\not\perp}$ for a $v' \in S$, then it must be proved that

$$\begin{aligned} &((\mathbf{WM} + \mathbf{T_c})^\star \bullet \varepsilon_\not\perp \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star)[s, v'] \\ &\quad \cdot (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star[v', s'] \\ &\leqslant ((\mathbf{WM} + \mathbf{T_c})^\star \bullet \varepsilon_\not\perp \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star)[s, s'] \quad . \end{aligned}$$

By Kleene algebra and by the definition of $\bullet$, it suffices to prove that

$$\begin{aligned} &\Big((\mathbf{WM} + \mathbf{T_c})^\star \bullet \varepsilon_\not\perp \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star \\ &\quad \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star\Big)[s, s'] \\ &\leqslant ((\mathbf{WM} + \mathbf{T_c})^\star \bullet \varepsilon_\not\perp \bullet (\mathbf{T}_\perp + \mathbf{WM} + (\mathbf{T_c} \bullet \mathbf{WM})^+ \bullet \varepsilon_\not\perp)^\star)[s, s'] \quad . \end{aligned}$$

This is trivial since $p^* = p^* p^*$ by Kleene algebra.

The proofs of (C.91) and (C.92) are similar to the proofs of (C.86).

The proofs of (C.93) are similar to the proofs of (C.85).

# Appendix D

# Proof of the Determinization of VPA (Theorem 4.8)

The construction used here is a simplification of Alur and Madhusudan's construction [1]. The deterministic automaton $\mathcal{A}'$ uses states of the form $T$ where $T \subseteq S \times S$ is a binary relation between states of $\mathcal{A}$. Intuitively, the range of the relation $T$ represents the current set of reachable states of $S$. This is the "standard" component of the usual subset construction of finite automata. On the other hand, the domain of $T$ represents the set of states reached *just after reading the last possibly pending call in the current run* (except for the initial state in which an identity function of $I$ is used). The component $T$ is used to postpone the evaluation of push actions on the stack until their associated pop action occurs (if there is such a pop action). Note that only the *evaluation* of the push action on the stack is postponed and not the *reading* of the call action. Note also that if there is no pop action associated to a push action, then the evaluation of this push action on the stack is not important since it is a pending call (it will never be used).

To understand the component $T$ more clearly, take a word $w = a_1 a_2 \dots a_i \dots a_j \dots a_k$ where $a_i$ is a call action and $a_j$ is its associated return action. Since $\mathcal{A}'$ is a deterministic pushdown automaton, there is one and only one run of $\mathcal{A}'$ on $w$. Suppose that we stop the run just after the action $a_j$. Name the state of $\mathcal{A}'$ just after the action $a_j$ by $T_{j+1}$. Each pair $(s, s') \in T_{j+1}$ represents that it is possible to start $\mathcal{A}$ in the state $s$ with an empty stack ($\perp$), run $\mathcal{A}$ on the affix $a_i \dots a_j$ and end the run in the state $s'$ of $\mathcal{A}$ with an empty stack ($\perp$).

We now express this construction formally. Let

$$\begin{array}{rcll} S' & := & 2^{S\times S}, & \\ \mathsf{ran}(T) & := & \{s' \in S \mid (\exists\ s \mid: (s,s') \in T)\} & \text{for } T \subseteq S \times S, \\ \mathsf{Id}_R & := & \{(s,s) \in S \times S \mid s \in R\} & \text{for } R \subseteq S. \end{array}$$

Construct the following deterministic visibly pushdown automaton:

$$\mathcal{A}' := (S', \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, S' \times \Sigma_{\mathbf{c}} \cup \{\bot\}, \delta', \{\mathsf{Id}_I\}, \{T \in S' \mid \mathsf{ran}(T) \cap F \neq \emptyset\})$$

where $\delta'$ is defined by the set of all the following transitions:

- a transition $(T, a, \lambda; T', \lambda)$ for each $a \in \Sigma_{\mathbf{i}}$ and $T \in S'$ where

$$T' := \{(s,s') \in S \times S \mid (\exists\ s'' \mid (s,s'') \in T : (s'', a, \lambda; s', \lambda) \in \delta)\}\ ;$$

- a transition $(T, a, \lambda; \mathsf{Id}_{R'}, (T, a))$ for each $a \in \Sigma_{\mathbf{c}}$ and $T \in S'$ where

$$R' := \{s' \in S \mid (\exists\ s, d \mid s \in \mathsf{ran}(T) \wedge d \in \Gamma : (s, a, \lambda; s', d) \in \delta)\}\ ;$$

- a transition $(T, a, \bot; T', \bot)$ for each $a \in \Sigma_{\mathbf{r}}$ and $T \in S'$ where

$$T' := \{(s,s') \in S \times S \mid (\exists\ s'' \mid (s,s'') \in T : (s'', a, \bot; s', \bot) \in \delta)\}\ ;$$

- a transition $(T, a, (T'', b); T', \lambda)$ for each $a \in \Sigma_{\mathbf{r}}$, $T \in S'$ and $(T'', b) \in S' \times \Sigma_{\mathbf{c}}$ where

$$T' := \{(s,s') \in S \times S \mid (\exists\ s'' \mid (s,s'') \in T'' : (s'', s') \in \mathsf{Update})\}$$

  and

$$\mathsf{Update} := \{(s,s') \in S \times S \mid (\exists s_1, s_2, d \mid (s_1,s_2) \in T \wedge d \in \Gamma : (s, b, \lambda; s_1, d) \in \delta \wedge (s_2, a, d; s', \lambda) \in \delta)\}\ .$$

We now show (4.23). To do this, we use a projection between states of $2^{S\times S}$ and states of $S$. The projection is in fact the composition of two simple projections $\mathbf{X}$ and $\mathbf{Y}$ expressed as matrices. Let $\mathbf{X}$ be a matrix of size $|2^{S\times S}| \times |S \times S|$ defined for each $T \subseteq S \times S$ and $s, s_2 \in S$ by

$$\mathbf{X}[T, (s_2, s)] := \begin{cases} 1 & \text{if } (s_2, s) \in T, \\ 0 & \text{otherwise.} \end{cases}$$

Let also $\mathbf{Y}$ be a matrix of size $|S \times S| \times |S|$ defined for each $s, s', s_2 \in S$ by

$$\mathbf{Y}[(s_2, s), s'] := \begin{cases} 1 & \text{if } s = s', \\ 0 & \text{otherwise.} \end{cases}$$

The projection $\mathbf{X} \bullet \mathbf{Y}$ is split in two because it helps a lot, for well-matched words, to have an "intermediate" level in which the component $T$ is "explicit".

The projection $\mathbf{X} \bullet \mathbf{Y}$ is useful since it suffices to prove that

$$\vdash \overrightarrow{I'}^t \bullet \mathbf{X} \bullet \mathbf{Y} = \overrightarrow{I}^t \ , \tag{D.1}$$

$$\vdash \overrightarrow{F'} = \mathbf{X} \bullet \mathbf{Y} \bullet \overrightarrow{F} \ , \tag{D.2}$$

$$\vdash \mathbf{T}'_{\mathsf{c}} \bullet \mathbf{X} \bullet \mathbf{Y} = \mathbf{X} \bullet \mathbf{Y} \bullet \mathbf{T}_{\mathsf{c}} \ , \tag{D.3}$$

$$\vdash \mathbf{T}'_{\perp} \bullet \mathbf{X} \bullet \mathbf{Y} = \mathbf{X} \bullet \mathbf{Y} \bullet \mathbf{T}_{\perp} \ , \tag{D.4}$$

$$\vdash \mathbf{WM}' \bullet \mathbf{X} \bullet \mathbf{Y} = \mathbf{X} \bullet \mathbf{Y} \bullet \mathbf{WM} \ , \tag{D.5}$$

and (4.23) follows easily,

$$\begin{array}{ll}
 & \overrightarrow{I'}^t \bullet (\mathbf{T}'_{\perp} + \mathbf{WM}')^\star \bullet (\mathbf{T}'_{\mathsf{c}} \bullet \mathbf{WM}')^\star \bullet \overrightarrow{F'} \\
= & \quad \{\!\{ \text{ Equation (D.2) } \}\!\} \\
 & \overrightarrow{I'}^t \bullet (\mathbf{T}'_{\perp} + \mathbf{WM}')^\star \bullet (\mathbf{T}'_{\mathsf{c}} \bullet \mathbf{WM}')^\star \bullet \mathbf{X} \bullet \mathbf{Y} \bullet \overrightarrow{F} \\
= & \quad \{\!\{ \text{ By (D.5) and (D.3), it is direct that } \mathbf{T}'_{\mathsf{c}} \bullet \mathbf{WM}' \bullet \mathbf{X} \bullet \mathbf{Y} = \mathbf{X} \bullet \mathbf{Y} \bullet \mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM}. \\
 & \quad \text{ \& Kleene algebra: Bisimulation rule } \}\!\} \\
 & \overrightarrow{I'}^t \bullet (\mathbf{T}'_{\perp} + \mathbf{WM}')^\star \bullet \mathbf{X} \bullet \mathbf{Y} \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \\
= & \quad \{\!\{ \text{ By (D.5), (D.4) and Kleene algebra, it is direct that } (\mathbf{T}'_{\perp} + \mathbf{WM}') \bullet \mathbf{X} \bullet \mathbf{Y} = \\
 & \quad \mathbf{X} \bullet \mathbf{Y} \bullet (\mathbf{T}_{\perp} + \mathbf{WM}). \text{ \& Kleene algebra: Bisimulation rule } \}\!\} \\
 & \overrightarrow{I'}^t \bullet \mathbf{X} \bullet \mathbf{Y} \bullet (\mathbf{T}_{\perp} + \mathbf{WM})^\star \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \\
= & \quad \{\!\{ \text{ Equation (D.1) } \}\!\} \\
 & \overrightarrow{I}^t \bullet (\mathbf{T}_{\perp} + \mathbf{WM})^\star \bullet (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F} \ .
\end{array}$$

So, it remains to prove (D.1) to (D.5).

## D.1 Proof of (D.1)

It suffices to show that for all $s \in S$,

$$(\overrightarrow{I'}^t \bullet \mathbf{X} \bullet \mathbf{Y})[1, s] = \overrightarrow{I}^t[1, s] \ .$$

$$\begin{array}{ll}
 & (\overrightarrow{I'}^t \bullet \mathbf{X} \bullet \mathbf{Y})[1, s] \\
= & \quad \{\!\{ \text{ Definition of } \bullet \ \}\!\}
\end{array}$$

$$
\begin{array}{ll}
 & (\sum T, s_2, s' \mid T \subseteq S \times S \wedge s_2, s' \in S : \overrightarrow{I'}^t[1,T] \cdot \mathbf{X}[T,(s_2,s')] \cdot \mathbf{Y}[(s_2,s'),s]) \\
= & \quad \{\!\{ \text{ Definition of } \mathbf{Y} \text{ and } \mathbf{X} \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } + \text{ \& One-point rule of } \sum \text{ } \}\!\} \\
 & (\sum T, s_2 \mid T \subseteq S \times S \wedge s_2 \in S \wedge (s_2,s) \in T : \overrightarrow{I'}^t[1,T]) \\
= & \quad \{\!\{ \text{ Definition of } \overrightarrow{I'} \text{: The only entry containing a 1 is exactly the entry } \mathsf{Id}_I \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } + \text{ } \}\!\} \\
 & (\sum T, s_2 \mid T = \mathsf{Id}_I \wedge s_2 \in S \wedge (s_2,s) \in T : 1) \\
= & \quad \{\!\{ \text{ One-point rule of } \sum \text{ } \}\!\} \\
 & (\sum s_2 \mid s_2 \in S \wedge (s_2,s) \in \mathsf{Id}_I : 1) \\
= & \quad \{\!\{ \text{ Definition of } \mathsf{Id}_I \text{ \& By definition, } I \subseteq S \text{ \& Definition of } \subseteq \text{ } \}\!\} \\
 & (\sum s_2 \mid s = s_2 \wedge s \in I : 1) \\
= & \quad \{\!\{ \text{ Definition of } \overrightarrow{I} \text{ } \}\!\} \\
 & (\sum s_2 \mid s_2 = s : \overrightarrow{I}^t[1,s]) \\
= & \quad \{\!\{ \text{ One-point rule of } \sum \text{ } \}\!\} \\
 & \overrightarrow{I}^t[1,s]
\end{array}
$$

## D.2 Proof of (D.2)

It suffices to show that for all $T \subseteq S \times S$,

$$\overrightarrow{F'}[T,1] = (\mathbf{X} \bullet \mathbf{Y} \bullet \overrightarrow{F})[T,1] \ .$$

$$
\begin{array}{ll}
 & (\mathbf{X} \bullet \mathbf{Y} \bullet \overrightarrow{F})[T,1] \\
= & \quad \{\!\{ \text{ Definition of } \bullet \text{ } \}\!\} \\
 & (\sum s, s_2, s' \mid s, s_2, s' \in S : \mathbf{X}[T,(s_2,s')] \cdot \mathbf{Y}[(s_2,s'),s] \cdot \overrightarrow{F}[s,1]) \\
= & \quad \{\!\{ \text{ Definition of } \mathbf{Y} \text{ and } \mathbf{X} \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } + \text{ \& One-point rule of } \sum \text{ } \}\!\} \\
 & (\sum s, s_2 \mid s, s_2 \in S \wedge (s_2,s) \in T : \overrightarrow{F}[s,1]) \\
= & \quad \{\!\{ \text{ Definition of } \overrightarrow{F} \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } + \text{ } \}\!\} \\
 & (\sum s, s_2 \mid s, s_2 \in S \wedge (s_2,s) \in T \wedge s \in F : 1) \\
= & \quad \{\!\{ \text{ Quantification laws \& Kleene algebra } \}\!\} \\
 & (\sum s \mid s \in S \wedge (\exists\ s_2 \mid : s_2 \in S \wedge (s_2,s) \in T) \wedge s \in F : 1) \\
= & \quad \{\!\{ \text{ Set theory \& Definition of } \mathsf{ran}(T) \text{ } \}\!\} \\
 & (\sum s \mid s \in \mathsf{ran}(T) \wedge s \in F : 1) \\
= & \quad \{\!\{ \text{ Definition of } \cap \text{ } \}\!\} \\
 & (\sum s \mid s \in \mathsf{ran}(T) \cap F : 1)
\end{array}
$$

$=$ {{ Kleene algebra and quantification laws: $(\sum s \mid s \in \mathsf{ran}(T) \cap F : 1) = 1$ if and only if $\mathsf{ran}(T) \cap F \neq \emptyset$ & Definition of $\overrightarrow{F'}$ }}
$\overrightarrow{F'}[T, 1]$

## D.3 Proof of (D.3)

It suffices to show that for all $T \subseteq S \times S$ and $s \in S$,

$$(\mathbf{T}'_{\mathsf{c}} \bullet \mathbf{X} \bullet \mathbf{Y})[T, s] = (\mathbf{X} \bullet \mathbf{Y} \bullet \mathbf{T}_{\mathsf{c}})[T, s] \ .$$

First, by Definition 4.3 of $\mathbf{T}'_{\mathsf{c}}$ and $\mathbf{T}_{\mathsf{c}}$, these two matrices can be expressed by the following summations:

$$\begin{aligned}\mathbf{T}_{\mathsf{c}} &= (\sum a, d \mid a \in \Sigma_{\mathbf{c}} \wedge d \in \Gamma : \mathbf{T}_{\mathsf{c}}^{a,d}),\\ \mathbf{T}'_{\mathsf{c}} &= (\sum a, b, T' \mid a, b \in \Sigma_{\mathbf{c}} \wedge T' \subseteq S \times S : (\mathbf{T}'_{\mathsf{c}})^{a,(T',b)}),\end{aligned}$$

where each matrix $\mathbf{T}_{\mathsf{c}}^{a,d}$ is of size $|S| \times |S|$ and contains only entries of the form 0 or $a$ and each matrix $(\mathbf{T}'_{\mathsf{c}})^{a,(T',b)}$ is of a size $|2^{S\times S}| \times |2^{S\times S}|$ and contains only entries of the form 0 or $a$. These matrices are a particular form of simple matrices as used by Kozen in [25]. Note also that, by definition of $\mathcal{A}'$, for all $T, T', T'' \subseteq S \times S$ and $a, b \in \Sigma_{\mathbf{c}}$,

$$(\mathbf{T}'_{\mathsf{c}})^{a,(T'',b)}[T, T'] = \begin{cases} a & \text{if } a = b \wedge T' = \mathsf{Id}_{\{t \in S \mid (\exists\ s', d \mid s' \in \mathsf{ran}(T) \wedge d \in \Gamma : (s', a, \lambda; t, d) \in \delta)\}} \wedge T = T'', \\ 0 & \text{otherwise,} \end{cases}$$

and for all $s', t \in S$,

$$\mathbf{T}_{\mathsf{c}}^{a,d}[s', t] = \begin{cases} a & \text{if } (s', a, \lambda; t, d) \in \delta, \\ 0 & \text{otherwise.} \end{cases}$$

Using these definitions, we are able to prove it.

$(\mathbf{T}'_{\mathsf{c}} \bullet \mathbf{X} \bullet \mathbf{Y})[T, s]$
$=$ {{ Definition of $\bullet$ }}
$(\sum T', s_2, s' \mid T' \subseteq S \times S \wedge s_2, s' \in S : \mathbf{T}'_{\mathsf{c}}[T, T'] \cdot \mathbf{X}[T', (s_2, s')] \cdot \mathbf{Y}[(s_2, s'), s])$
$=$ {{ Definition of $\mathbf{Y}$ and $\mathbf{X}$ & Zero of $\cdot$ & Identity of $\cdot$ and $+$ & One-point rule of $\sum$ }}
$(\sum T', s_2 \mid T' \subseteq S \times S \wedge s_2 \in S \wedge (s_2, s) \in T' : \mathbf{T}'_{\mathsf{c}}[T, T'])$
$=$ {{ Definition of $\mathbf{T}'_{\mathsf{c}}$ }}

$(\sum T', s_2 \mid T' \subseteq S \times S \wedge s_2 \in S \wedge (s_2, s) \in T' : (\sum a, b, T'' \mid a, b \in \Sigma_{\mathbf{c}} \wedge T'' \subseteq S \times S : (\mathbf{T}'_{\mathsf{c}})^{a,(T'',b)})[T, T'])$

$=$ {{ Definition of $+$ }}

$(\sum T', s_2 \mid T' \subseteq S \times S \wedge s_2 \in S \wedge (s_2, s) \in T' : (\sum a, b, T'' \mid a, b \in \Sigma_{\mathbf{c}} \wedge T'' \subseteq S \times S : (\mathbf{T}'_{\mathsf{c}})^{a,(T'',b)}[T, T']))$

$=$ {{ Definition of $(\mathbf{T}'_{\mathsf{c}})^{a,(T'',b)}[T, T']$ & Identity of $+$ }}

$(\sum T', s_2 \mid T' \subseteq S \times S \wedge s_2 \in S \wedge (s_2, s) \in T' : (\sum a, b, T'' \mid a, b \in \Sigma_{\mathbf{c}} \wedge T'' \subseteq S \times S \wedge a = b \wedge T' = \mathsf{Id}_{\{t \in S \mid (\exists\ s', d \mid s' \in \mathsf{ran}(T) \wedge d \in \Gamma : (s', a, \lambda; t, d) \in \delta)\}} \wedge T = T'' : a))$

$=$ {{ One-point rule of $\sum$ }}

$(\sum T', s_2 \mid T' \subseteq S \times S \wedge s_2 \in S \wedge (s_2, s) \in T' : (\sum a \mid a \in \Sigma_{\mathbf{c}} \wedge T' = \mathsf{Id}_{\{t \in S \mid (\exists\ s', d \mid s' \in \mathsf{ran}(T) \wedge d \in \Gamma : (s', a, \lambda; t, d) \in \delta)\}} : a))$

$=$ {{ Nesting & One-point rule of $\sum$ }}

$(\sum s_2, a \mid s_2 \in S \wedge (s_2, s) \in \mathsf{Id}_{\{t \in S \mid (\exists\ s', d \mid s' \in \mathsf{ran}(T) \wedge d \in \Gamma : (s', a, \lambda; t, d) \in \delta)\}} \wedge a \in \Sigma_{\mathbf{c}} : a)$

$=$ {{ Set theory }}

$(\sum s_2, a \mid s_2 \in S \wedge s_2 = s \wedge s \in S \wedge (\exists\ s', d \mid s' \in \mathsf{ran}(T) \wedge d \in \Gamma : (s', a, \lambda; s, d) \in \delta) \wedge a \in \Sigma_{\mathbf{c}} : a)$

$=$ {{ Hypothesis: $s \in S$ & One-point rule of $\sum$ }}

$(\sum a \mid a \in \Sigma_{\mathbf{c}} \wedge (\exists\ s', d \mid s' \in \mathsf{ran}(T) \wedge d \in \Gamma : (s', a, \lambda; s, d) \in \delta) : a)$

$=$ {{ Quantification laws & Kleene algebra }}

$(\sum s', a, d \mid s' \in \mathsf{ran}(T) \wedge a \in \Sigma_{\mathbf{c}} \wedge d \in \Gamma \wedge (s', a, \lambda; s, d) \in \delta : a)$

$=$ {{ Definition of $\mathsf{ran}(T)$ & Set theory }}

$(\sum s', a, d \mid s' \in S \wedge (\exists\ s_2 \mid s_2 \in S : (s_2, s') \in T) \wedge a \in \Sigma_{\mathbf{c}} \wedge d \in \Gamma \wedge (s', a, \lambda; s, d) \in \delta : a)$

$=$ {{ Quantification laws & Kleene algebra }}

$(\sum s_2, s', a, d \mid s_2, s' \in S \wedge (s_2, s') \in T \wedge a \in \Sigma_{\mathbf{c}} \wedge d \in \Gamma \wedge (s', a, \lambda; s, d) \in \delta : a)$

$=$ {{ Nesting & Definition of $\mathbf{T}_{\mathsf{c}}^{a,d}[s', s]$ & Identity of $+$ }}

$(\sum s_2, s' \mid s_2, s' \in S \wedge (s_2, s') \in T : (\sum a, d \mid a \in \Sigma_{\mathbf{c}} \wedge d \in \Gamma : \mathbf{T}_{\mathsf{c}}^{a,d}[s', s]))$

$=$ {{ Definition of $+$ }}

$(\sum s_2, s' \mid s_2, s' \in S \wedge (s_2, s') \in T : (\sum a, d \mid a \in \Sigma_{\mathbf{c}} \wedge d \in \Gamma : \mathbf{T}_{\mathsf{c}}^{a,d})[s', s])$

$=$ {{ Definition of $\mathbf{T}_{\mathsf{c}}$ }}

$(\sum s_2, s' \mid s_2, s' \in S \wedge (s_2, s') \in T : \mathbf{T}_{\mathsf{c}}[s', s])$

$=$ {{ Definition of $\mathbf{Y}$ and $\mathbf{X}$ & Zero of $\cdot$ & Identity of $\cdot$ and $+$ & One-point rule of $\sum$ }}

$(\sum s_2, s', s'' \mid s_2, s', s'' \in S : \mathbf{X}[T, (s_2, s')] \cdot \mathbf{Y}[(s_2, s'), s''] \cdot \mathbf{T}_{\mathsf{c}}[s'', s])$

$=$ {{ Definition of $\bullet$ }}

$(\mathbf{X} \bullet \mathbf{Y} \bullet \mathbf{T}_{\mathsf{c}})[T, s]$

## D.4 Proof of (D.4)

It suffices to show that for all $T \subseteq S \times S$ and $s \in S$,

$$(\mathbf{T}'_\perp \bullet \mathbf{X} \bullet \mathbf{Y})[T, s] = (\mathbf{X} \bullet \mathbf{Y} \bullet \mathbf{T}_\perp)[T, s] \ .$$

First, by Definition 4.3 of $\mathbf{T}'_\perp$ and $\mathbf{T}_\perp$, these two matrices can be expressed by the following summations:

$$\mathbf{T}_\perp = (\sum\ a \mid a \in \Sigma_{\mathbf{r}} : \mathbf{T}^a_\perp) \ ,$$
$$\mathbf{T}'_\perp = (\sum\ a \mid a \in \Sigma_{\mathbf{r}} : (\mathbf{T}'_\perp)^a) \ ,$$

where each matrix $\mathbf{T}^a_\perp$ is of size $|S| \times |S|$ and contains only entries of the form 0 or $a$ and each matrix $(\mathbf{T}'_\perp)^a$ is of a size $|2^{S\times S}| \times |2^{S\times S}|$ and contains only entries of the form 0 or $a$. These matrices are a particular form of simple matrices as used by Kozen in [25]. Note also that, by definition of $\mathcal{A}'$, for all $T, T' \subseteq S \times S$ and $a \in \Sigma_{\mathbf{r}}$,

$$(\mathbf{T}'_\perp)^a[T, T'] = \begin{cases} a & \text{if } T' = \{(t, s') \in S \times S \mid (\exists\ s'' \mid (t, s'') \in T : (s'', a, \perp; s', \perp) \in \delta)\}, \\ 0 & \text{otherwise,} \end{cases}$$

and for all $s', t \in S$,

$$\mathbf{T}^a_\perp d[s', t] = \begin{cases} a & \text{if } (s', a, \perp; t, \perp) \in \delta, \\ 0 & \text{otherwise.} \end{cases}$$

Using these definitions, we are able to prove it.

$(\mathbf{T}'_\perp \bullet \mathbf{X} \bullet \mathbf{Y})[T, s]$
$=$ {{ Definition of $\bullet$ }}
$(\sum\ T', s_2, s' \mid T' \subseteq S \times S \wedge s_2, s' \in S : \mathbf{T}'_\perp[T, T'] \cdot \mathbf{X}[T', (s_2, s')] \cdot \mathbf{Y}[(s_2, s'), s])$
$=$ {{ Definition of $\mathbf{Y}$ and $\mathbf{X}$ & Zero of $\cdot$ & Identity of $\cdot$ and $+$ & One-point rule of $\sum$ }}
$(\sum\ T', s_2 \mid T' \subseteq S \times S \wedge s_2 \in S \wedge (s_2, s) \in T' : \mathbf{T}'_\perp[T, T'])$
$=$ {{ Definition of $\mathbf{T}'_\perp$ }}
$(\sum\ T', s_2 \mid T' \subseteq S \times S \wedge s_2 \in S \wedge (s_2, s) \in T' : (\sum\ a \mid a \in \Sigma_{\mathbf{r}} : \mathbf{T}^a_\perp)[T, T'])$
$=$ {{ Definition of $+$ }}
$(\sum\ T', s_2 \mid T' \subseteq S \times S \wedge s_2 \in S \wedge (s_2, s) \in T' : (\sum\ a \mid a \in \Sigma_{\mathbf{r}} : \mathbf{T}^a_\perp[T, T']))$
$=$ {{ Definition of $(\mathbf{T}'_\perp)^a[T, T']$ & Identity of $+$ }}
$(\sum\ T', s_2 \mid T' \subseteq S \times S \wedge s_2 \in S \wedge (s_2, s) \in T' : (\sum\ a \mid a \in \Sigma_{\mathbf{r}} \wedge T' = \{(t, s') \in S \times S \mid (\exists\ s'' \mid (t, s'') \in T : (s'', a, \perp; s', \perp) \in \delta)\} : a))$
$=$ {{ Nesting & One-point rule of $\sum$ }}

$(\sum\ s_2, a \mid a \in \Sigma_{\mathbf{r}} \wedge s_2 \in S \wedge (s_2, s) \in \{(t, s') \in S \times S \mid (\exists\ s'' \mid (t, s'') \in T : (s'', a, \bot; s', \bot) \in \delta)\} : a)$

$=$ $\{\!\{$ Set theory $\}\!\}$

$(\sum\ s_2, a \mid a \in \Sigma_{\mathbf{r}} \wedge s_2 \in S \wedge (s_2, s) \in S \times S \wedge (\exists\ s'' \mid (s_2, s'') \in T : (s'', a, \bot; s, \bot) \in \delta) : a)$

$=$ $\{\!\{$ By definition, $s \in S \wedge s_2 \in S \Rightarrow (s_2, s) \in S \times S$ & Definition of $\Rightarrow$ $\}\!\}$

$(\sum\ s_2, a \mid a \in \Sigma_{\mathbf{r}} \wedge s_2 \in S \wedge (\exists\ s'' \mid (s_2, s'') \in T : (s'', a, \bot; s, \bot) \in \delta) : a)$

$=$ $\{\!\{$ Kleene algebra & Quantification laws & By definition, $(s_2, s'') \in T \Rightarrow s'' \in S$ & Definition of $\Rightarrow$ $\}\!\}$

$(\sum\ s_2, s'', a \mid a \in \Sigma_{\mathbf{r}} \wedge s_2, s'' \in S \wedge (s_2, s'') \in T \wedge (s'', a, \bot; s, \bot) \in \delta : a)$

$=$ $\{\!\{$ Nesting & Definition of $\mathbf{T}^a_\bot[s'', s]$ & Identity of $+$ $\}\!\}$

$(\sum\ s_2, s'' \mid s_2, s'' \in S \wedge (s_2, s'') \in T : (\sum\ a \mid a \in \Sigma_{\mathbf{r}} : \mathbf{T}^a_\bot[s'', s]))$

$=$ $\{\!\{$ Definition of $\mathbf{+}$ $\}\!\}$

$(\sum\ s_2, s'' \mid s_2, s'' \in S \wedge (s_2, s'') \in T : (\sum\ a \mid a \in \Sigma_{\mathbf{r}} : \mathbf{T}^a_\bot)[s'', s])$

$=$ $\{\!\{$ Definition of $\mathbf{T}_\bot$ $\}\!\}$

$(\sum\ s_2, s'' \mid s_2, s'' \in S \wedge (s_2, s'') \in T : \mathbf{T}_\bot[s'', s])$

$=$ $\{\!\{$ Definition of $\mathbf{Y}$ and $\mathbf{X}$ & Zero of $\cdot$ & Identity of $\cdot$ and $+$ & One-point rule of $\sum$ $\}\!\}$

$(\sum\ s_2, s', s'' \mid s_2, s', s'' \in S : \mathbf{X}[T, (s_2, s')] \cdot \mathbf{Y}[(s_2, s'), s''] \cdot \mathbf{T}_\bot[s'', s])$

$=$ $\{\!\{$ Definition of $\bullet$ $\}\!\}$

$(\mathbf{X} \bullet \mathbf{Y} \bullet \mathbf{T}_\bot)[T, s]$

## D.5 Proof of (D.5)

It suffices to show that for all $T \subseteq S \times S$ and $s \in S$,

$$(\mathbf{WM}' \bullet \mathbf{X} \bullet \mathbf{Y})[T, s] = (\mathbf{X} \bullet \mathbf{Y} \bullet \mathbf{WM})[T, s] \ .$$

We prove it.

$(\mathbf{WM}' \bullet \mathbf{X} \bullet \mathbf{Y})[T, s]$

$=$ $\{\!\{$ Definition of $\bullet$ $\}\!\}$

$(\sum\ s_2, s' \mid s_2, s' \in S : (\sum\ T' \mid T' \subseteq S \times S : \mathbf{WM}'[T, T'] \cdot \mathbf{X}[T', (s_2, s')]) \cdot \mathbf{Y}[(s_2, s'), s])$

$=$ $\{\!\{$ Definition of $\mathbf{Y}$ & Zero of $\cdot$ & Identity of $\cdot$ and $+$ & One-point rule of $\sum$ $\}\!\}$

$(\sum\ s_2 \mid s_2 \in S : (\sum\ T' \mid T' \subseteq S \times S : \mathbf{WM}'[T, T'] \cdot \mathbf{X}[T', (s_2, s)]))$

$=$ $\{\!\{$ Definition of $\mathbf{WM}'$ $\}\!\}$

$$
\begin{array}{ll}
 & (\sum\, s_2 \mid s_2 \in S : (\sum\, T' \mid T' \subseteq S \times S : (\!|_T\, \mathcal{B}'\, |\!)^{T'} \cdot \mathbf{X}[T', (s_2, s)])) \\
= & \quad \{\!\{ \text{ Equation (D.6) (see below) } \}\!\} \\
 & (\sum\, s_2 \mid s_2 \in S : (\sum\, s' \mid s' \in S : \mathbf{X}[T, (s_2, s')] \cdot (\!|_{s'}\, \mathcal{B}\, |\!)^{s})) \\
= & \quad \{\!\{ \text{ Definition of } \mathbf{WM} \}\!\} \\
 & (\sum\, s_2 \mid s_2 \in S : (\sum\, s' \mid s' \in S : \mathbf{X}[T, (s_2, s')] \cdot \mathbf{WM}[s', s])) \\
= & \quad \{\!\{ \text{ Kleene algebra \& Quantification laws } \}\!\} \\
 & (\sum\, s' \mid s' \in S : (\sum\, s_2 \mid s_2 \in S : \mathbf{X}[T, (s_2, s')]) \cdot \mathbf{WM}[s', s]) \\
= & \quad \{\!\{ \text{ Definition of } \mathbf{Y} \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } + \text{ \& One-point rule of } \\
 & \qquad \sum \}\!\} \\
 & (\sum\, s' \mid s' \in S : (\sum\, s_2, s'' \mid s_2, s'' \in S : \mathbf{X}[T, (s_2, s'')] \cdot \mathbf{Y}[(s_2, s''), s']) \cdot \mathbf{WM}[s', s]) \\
= & \quad \{\!\{ \text{ Definition of } \bullet \}\!\} \\
 & (\mathbf{X} \bullet \mathbf{Y} \bullet \mathbf{WM})[T, s]
\end{array}
$$

To finish the proof of (D.5), it remains to prove that for all $T \subseteq S \times S$ and $s, s_2 \in S$,

$$
\begin{array}{l}
\vdash (\sum\, T' \mid T' \subseteq S \times S : \underset{T}{(\!|}\, \mathcal{B}'\, \overset{T'}{|\!)} \cdot \mathbf{X}[T', (s_2, s)]) \\
\qquad = \ (\sum\, s' \mid s' \in S : \mathbf{X}[T, (s_2, s')] \cdot \underset{s'}{(\!|}\, \mathcal{B}\, \overset{s}{|\!)}) \ .
\end{array}
\tag{D.6}
$$

We first explicitly describe the list of blocks of $\mathcal{B}'$ with respect to $\mathcal{B}$:

a. The set of labels $2^{S \times S}$ is defined.

b. For all $T \subseteq S \times S$,

$$\underset{T}{[}\, 1 \,\overset{T}{]} \in (\mathcal{B}')^1 \ .$$

c. For all internal actions $a \in \Sigma_{\mathbf{i}}$ and $T, T' \subseteq S \times S$,

$$\underset{T}{[}\, a \,\overset{T'}{]} \in (\mathcal{B}')^1 \Leftrightarrow T' = \{(s, s') \in S \times S \mid (\exists\, s'' \mid (s, s'') \in T : \underset{s''}{[}\, a \,\overset{s'}{]} \in \mathcal{B}^1)\} \ .$$

d. For all call actions $c \in \Sigma_{\mathbf{c}}$, return actions $r \in \Sigma_{\mathbf{r}}$ and $T_1, T_2, T_3, T_4 \subseteq S \times S$,

$$
\begin{array}{l}
\underset{T_1}{[}\, c \underset{T_2}{\downarrow} \overset{T_3}{\uparrow} r \,\overset{T_4}{]} \in (\mathcal{B}')^2 \\
\Leftrightarrow \ T_2 = \mathsf{Id}_{\{s' \in S \mid (\exists\ s, d \mid s \in \mathsf{ran}(T_1) \wedge d \in \Gamma : (s, c, \lambda; s', d) \in \delta)\}} \\
\quad \wedge \ T_4 = \{(s, s') \in S \times S \mid \\
\qquad\qquad (\exists\, s'', s_1, s_2 \mid (s, s'') \in T_1 \wedge (s_1, s_2) \in T_3 : \underset{s''}{[}\, c \underset{s_1}{\downarrow} \overset{s_2}{\uparrow} r \,\overset{s'}{]} \in \mathcal{B}^2)\} \ .
\end{array}
$$

### D.5.1 Proof of the Case $\geqslant$ of (D.6)

We now show the case $\geqslant$ of (D.6). By Kleene algebra, it suffices to show independently that for all $T \subseteq S \times S$ and $s, s_2, s' \in S$,

$$\mathbf{X}[T, (s_2, s')] \cdot (\!|_{s'} \mathcal{B} |\!)^{s} \leqslant (\sum T' \mid T' \subseteq S \times S : (\!|_{T} \mathcal{B}' |\!)^{T'} \cdot \mathbf{X}[T', (s_2, s)]) \ .$$

By (3.16), it suffices to show that

- for all $s'_1 \in S$, $T_1 \subseteq S \times S$ and $[_{s_1}\, m\, ]^{s_4} \in \mathcal{B}^1$,

$$\mathbf{X}[(T_1, (s'_1, s_1))] \cdot m \leqslant (\sum T_4 \mid [\,_{T_1}^{\ } m\, ]^{T_4} \in (\mathcal{B}')^1 : m \cdot \mathbf{X}[T_4, (s'_1, s_4)]) \ , \tag{D.7}$$

- for all $s'_1 \in S$, $T_1 \subseteq S \times S$ and $[_{s_1}\, c{\downarrow_{s_2}} {\uparrow^{s_3}}\, r\, ]^{s_4} \in \mathcal{B}^2$,

$$\begin{aligned} \mathbf{X}[T_1, (s'_1, s_1)] \cdot c \leqslant (\sum T_2, T_3, T_4 \mid [_{T_1}\, c{\downarrow_{T_2}} {\uparrow^{T_3}}\, r\, ]^{T_4} \in (\mathcal{B}')^2 : \\ \mathbf{X}[T_1, (s'_1, s_1)] \cdot c \cdot \mathbf{X}[T_2, (s_2, s_2)] \cdot \mathbf{X}[T_2, (s_2, s_2)])) \ , \end{aligned} \tag{D.8}$$

- for all $s'_1 \in S$, $T_1, T_2, T'_3 \subseteq S \times S$ and $[_{s_1}\, c{\downarrow_{s_2}} {\uparrow^{s_3}}\, r\, ]^{s_4} \in \mathcal{B}^2$,

$$\begin{aligned} &(\sum T_3, T_4 \mid [_{T_1}\, c{\downarrow_{T_2}} {\uparrow^{T_3}}\, r\, ]^{T_4} \in (\mathcal{B}')^2 : \\ &\qquad \mathbf{X}[T_1, (s'_1, s_1)] \cdot c \cdot \mathbf{X}[T_2, (s_2, s_2)] \cdot (\!|_{T_2} \mathcal{B}' |\!)^{T'_3} \cdot \mathbf{X}[T'_3, (s_2, s_3)] \cdot r) \\ &\leqslant \ (\sum T_4 \mid [_{T_1}\, c{\downarrow_{T_2}} {\uparrow^{T'_3}}\, r\, ]^{T_4} \in (\mathcal{B}')^2 : c \cdot (\!|_{T_2} \mathcal{B}' |\!)^{T'_3} \cdot r \cdot \mathbf{X}[T_4, (s'_1, s_4)]) \ . \end{aligned} \tag{D.9}$$

We first show (D.7). To do this, we show a stronger result: for all $T_1 \subseteq S \times S$, $s'_1, s_4 \in S$ and $m \in \Sigma_{\mathbf{i}} \cup \{0, 1\}$,

$$\begin{aligned} &(\sum s''_1 \mid [\,_{s''_1}^{\ } m\, ]^{s_4} \in \mathcal{B}^1 : \mathbf{X}[(T_1, (s'_1, s''_1))] \cdot m) \\ &= \ (\sum T_4 \mid [\,_{T_1}^{\ } m\, ]^{T_4} \in (\mathcal{B}')^1 : m \cdot \mathbf{X}[T_4, (s'_1, s_4)]) \ . \end{aligned} \tag{D.10}$$

Let us prove this by case analysis on $m$. For the case $m = 0$, by the definition of $\mathcal{B}$ and $\mathcal{B}'$, there is no unary block containing 0 as operand. So, the empty range rule can be applied for both sides of the equation, yielding the same result.

For the case $m = 1$, by condition B of the definition of $\mathcal{B}$, $s''_1 = s_4$. Also, by condition b of the definition of $\mathcal{B}'$, $T_4 = T_1$. So, using the one-point rule of $\sum$ for both sides of the equation, we have to show that

$$\mathbf{X}[(T_1, (s'_1, s_4))] \cdot 1 = 1 \cdot \mathbf{X}[T_1, (s'_1, s_4)] \ .$$

This is trivial by the identity of $\cdot$.

For the case $m \in \Sigma_{\mathbf{i}}$,

$$
\begin{array}{ll}
 & (\sum T_4 \mid [_{T_1}\, m\, ]^{T_4} \in (\mathcal{B}')^1 : m \cdot \mathbf{X}[T_4, (s_1', s_4)]) \\
= & \quad \{\!\{ \text{ Definition of } \mathbf{X} \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } + \ \}\!\} \\
 & (\sum T_4 \mid [_{T_1}\, m\, ]^{T_4} \in (\mathcal{B}')^1 \wedge (s_1', s_4) \in T_4 : m) \\
= & \quad \{\!\{ \text{ Hypotheses: } m \in \Sigma_{\mathbf{i}} \text{ and } \mathcal{B}' \text{ is deterministic \& Definition of } \mathcal{B}', \text{ condition c } \}\!\} \\
 & (\sum T_4 \mid T_4 = \{(s, s') \in S \times S \mid (\exists\ s_1'' \mid (s, s_1'') \in T_1 : [_{s_1''}\, m\, ]^{s'} \in \mathcal{B}^1)\} \wedge (s_1', s_4) \in T_4 : m) \\
= & \quad \{\!\{ \text{ Quantification laws \& Kleene algebra \& Set theory } \}\!\} \\
 & (\sum s_1'' \mid (s_1', s_1'') \in T_1 \wedge [_{s_1''}\, m\, ]^{s_4} \in \mathcal{B}^1 : m) \\
= & \quad \{\!\{ \text{ Definition of } \mathbf{X} \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } + \ \}\!\} \\
 & (\sum s_1'' \mid [_{s_1''}\, m\, ]^{s_4} \in \mathcal{B}^1 : \mathbf{X}[(T_1, (s_1', s_1''))] \cdot m)\ .
\end{array}
$$

Then, inequation (D.7) is proved by noting that, by hypothesis $[_{s_1}\, m\, ]^{s_4} \in \mathcal{B}^1$ and Kleene algebra,

$$\mathbf{X}[(T_1, (s_1', s_1))] \cdot m \leqslant (\sum s_1'' \mid [\!\!\underset{s_1''}{}\!\!\, m\, ]^{s_4} \in \mathcal{B}^1 : \mathbf{X}[(T_1, (s_1', s_1''))] \cdot m)\ ,$$

and equation (D.10).

We now show (D.8). In the case that $(s_1', s_1) \notin T_1$, the proof is trivial since $\mathbf{X}[T_1, (s_1', s_1)] = 0$. So, suppose that $(s_1', s_1) \in T_1$. To prove (D.8), by the definition of $\mathbf{X}$ and identity of $\cdot$ and $+$, it suffices to show that

$$c \leqslant (\sum T_2, T_3, T_4 \mid [\!\!\underset{T_1}{}\!\!\, c \downarrow_{T_2} \uparrow^{T_3} r\, ]^{T_4} \in (\mathcal{B}')^2 \wedge (s_2, s_2) \in T_2 : c)\ .$$

To prove the previous inequation, by Kleene algebra, it suffices to prove that there exists at least one $T_2, T_3, T_4 \subseteq S \times S$ such that $[_{T_1}\, c \downarrow_{T_2} \uparrow^{T_3} r\, ]^{T_4} \in (\mathcal{B}')^2$ and $(s_2, s_2) \in T_2$.

First, since $(s_1', s_1) \in T_1$ and $[_{s_1}\, c \downarrow_{s_2} \uparrow^{s_3} r\, ]^{s_4} \in \mathcal{B}^2$ by hypothesis, it is easy to see that $(\exists\ s, d \mid s \in \mathsf{ran}(T_1) \wedge d \in \Gamma : (s, c, \lambda; s_2, d) \in \delta)$. In other words,

$$(s_2, s_2) \in \mathsf{Id}_{\{s' \in S \mid (\exists\ s, d \mid s \in \mathsf{ran}(T_1) \wedge d \in \Gamma : (s, c, \lambda; s', d) \in \delta)\}}\ . \tag{D.11}$$

We will see that the set $\mathsf{Id}_{\{s' \in S \mid (\exists\ s, d \mid s \in \mathsf{ran}(T_1) \wedge d \in \Gamma : (s, c, \lambda; s', d) \in \delta)\}}$ is the wanted $T_2$.

Second, by the fact that $\mathcal{B}'$ is deterministic and the definition of $\mathcal{B}'$, condition d, for every $T'_1, T_3 \subseteq S \times S$, there exists one and only one $T_2, T_4 \subseteq S \times S$ such that $\lfloor_{T'_1} c\downarrow_{T_2} \uparrow^{T_3} r \rfloor^{T_4} \in (\mathcal{B}')^2$. In particular, if $T'_1$ is $T_1$, then $T_2$ is

$$\mathsf{Id}_{\{s' \in S \mid (\exists\ s,d \mid s \in \mathsf{ran}(T_1) \wedge d \in \Gamma : (s,c,\lambda;s',d) \in \delta)\}} \ .$$

By the fact that $2^{S\times S}$ is nonempty and by the previous reasonings, we have that there exists at least one $T_2, T_3, T_4 \subseteq S \times S$ such that $\lfloor_{T_1} c\downarrow_{T_2} \uparrow^{T_3} r \rfloor^{T_4} \in (\mathcal{B}')^2$ and $T_2 = \mathsf{Id}_{\{s' \in S \mid (\exists\ s,d \mid s \in \mathsf{ran}(T_1) \wedge d \in \Gamma : (s,c,\lambda;s',d) \in \delta)\}}$. Then, the result follows from (D.11).

We now show (D.9). By Kleene algebra and the definition of $\mathbf{X}$, it suffices to prove that, for all $s'_1 \in S$, $T_1, T_2, T_3, T'_3, T_4 \subseteq S \times S$ and $\lfloor_{s_1} c\downarrow_{s_2} \uparrow^{s_3} r \rfloor^{s_4} \in \mathcal{B}^2$ such that $\lfloor_{T_1} c\downarrow_{T_2} \uparrow^{T_3} r \rfloor^{T_4} \in (\mathcal{B}')^2$, $(s'_1, s_1) \in T_1$, $(s_2, s_2) \in T_2$ and $(s_2, s_3) \in T'_3$,

$$c \cdot \underset{T_2}{(\!|}\, \mathcal{B}' \,\overset{T'_3}{|\!)} \cdot r \leqslant (\textstyle\sum T'_4 \mid \lfloor \underset{T_1}{c\downarrow} \overset{T'_3}{\uparrow} r \rfloor^{T'_4} \in (\mathcal{B}')^2 \wedge (s'_1, s_4) \in T'_4 : c \cdot \underset{T_2}{(\!|}\, \mathcal{B}' \,\overset{T'_3}{|\!)} \cdot r) \ .$$

By Kleene algebra, it suffices to prove that there exists a $T'_4$ such that $\lfloor_{T_1} c\downarrow_{T_2} \uparrow^{T'_3} r \rfloor^{T'_4} \in (\mathcal{B}')^2$ and $(s'_1, s_4) \in T'_4$. We show that $T'_4$ is exactly

$$\{(s, s') \in S \times S \mid (\exists\ s''_1, s'_2, s'_3 \mid (s, s''_1) \in T_1 \wedge (s'_2, s'_3) \in T'_3 : \lfloor \underset{s''_1}{c\downarrow} \overset{s'_3}{\underset{s'_2}{\uparrow}} r \rfloor^{s'} \in \mathcal{B}^2)\} \ .$$

To show that $\lfloor_{T_1} c\downarrow_{T_2} \uparrow^{T'_3} r \rfloor^{T'_4} \in (\mathcal{B}')^2$, by the definition of $\mathcal{B}'$, it suffices to prove that

$$(T_1, c, \lambda; T_2, (T_1, c)) \in \delta' \qquad \wedge \qquad (T'_3, r, (T_1, c); T'_4, \lambda) \in \delta' \ .$$

The formula $(T_1, c, \lambda; T_2, (T_1, c)) \in \delta'$ is immediate from the hypothesis $\lfloor_{T_1} c\downarrow_{T_2} \uparrow^{T_3} r \rfloor^{T_4} \in (\mathcal{B}')^2$ and the definition of $\mathcal{B}'$ which state together that

$$(T_1, c, \lambda; T_2, (T_1, c)) \in \delta' \qquad \wedge \qquad (T_3, r, (T_1, c); T_4, \lambda) \in \delta' \ .$$

We now prove that $(T'_3, r, (T_1, c); T'_4, \lambda) \in \delta'$. This is immediate from the definition of $T'_4$ and the definition of $\mathcal{B}'$ of page 245.

We now show that $(s'_1, s_4) \in T'_4$. By the definition of $T'_4$ and set theory, it suffices to prove that

$$(\exists\ s''_1, s'_2, s'_3 \mid (s'_1, s''_1) \in T_1 \wedge (s'_2, s'_3) \in T'_3 : \lfloor \underset{s''_1}{c\downarrow} \overset{s'_3}{\underset{s'_2}{\uparrow}} r \rfloor^{s_4} \in \mathcal{B}^2) \ .$$

This is easily proved by using $s_1$ as $s''_1$, $s_2$ as $s'_2$ and $s_3$ as $s'_3$ along with the hypotheses $(s'_1, s_1) \in T_1$, $(s_2, s_3) \in T'_3$ and $\lfloor_{s_1} c\downarrow_{s_2} \uparrow^{s_3} r \rfloor^{s_4} \in \mathcal{B}^2$.

### D.5.2 Proof of the Case $\leqslant$ of (D.6)

We now show the case $\leqslant$ of (D.6). By Kleene algebra, it suffices to show independently that for all $T, T' \subseteq S \times S$ and $s, s_2 \in S$,

$$\left(\!\!\left| \mathcal{B}' \right|\!\!\right)_{T}^{T'} \cdot \mathbf{X}[T', (s_2, s)] \leqslant \left(\sum s' \mid s' \in S : \mathbf{X}[T, (s_2, s')] \cdot \left(\!\!\left| \mathcal{B} \right|\!\!\right)_{s'}^{s}\right) .$$

By definition of $\mathbf{X}$, the only way that $\mathbf{X}[T', (s_2, s)] = 1$ is when $(s_2, s) \in T'$. Otherwise, $\mathbf{X}[T', (s_2, s)] = 0$ and the proof is trivial. So, suppose $(s_2, s) \in T'$. Then, any expression $\left(\!\!\left|_{T} \mathcal{B}' \right|\!\!\right)^{T'}$ that will be used is such that $T' \neq \emptyset$. We first remove from $\mathcal{B}'$ every unary and binary blocks containing $\emptyset$ as their ending label since they cannot reach $T'$. Let $\mathcal{B}''$ be such a list (the list $\mathcal{B}'$ without every unary and binary blocks containing $\emptyset$ as their ending label). We show that

$$\vdash \left(\!\!\left| \mathcal{B}' \right|\!\!\right)_{T}^{T'} = \left(\!\!\left| \mathcal{B}'' \right|\!\!\right)_{T}^{T'} \tag{D.12}$$

when $T' \neq \emptyset$. The case $\geqslant$ is direct by (3.26) and (3.30). For the case $\leqslant$, we first note that any block containing a label $\emptyset$ in any position (starting, ending, call or return) also has an ending label $\emptyset$. In fact, by the definition of $\mathcal{B}'$, it is easy to see that

- if a unary block of $\mathcal{B}'$ is of the form $\left[_{\emptyset}\, a \,\right]^{T_4}$, then $T_4 = \emptyset$;
- if a binary block of $\mathcal{B}'$ is of the form $\left[_{T_1}\, c \downarrow_{\emptyset} \uparrow^{T_3} r \,\right]^{T_4}$, then $T_4 = \emptyset$;
- if a binary block of $\mathcal{B}'$ is of the form $\left[_{\emptyset}\, c \downarrow_{T_2} \uparrow^{T_3} r \,\right]^{T_4}$, then $T_2 = \emptyset$ and so $T_4 = \emptyset$;
- if a binary block of $\mathcal{B}'$ is of the form $\left[_{T_1}\, c \downarrow_{T_2} \uparrow^{\emptyset} r \,\right]^{T_4}$, then $T_4 = \emptyset$.

So, the set $\mathsf{B}^{*}_{\mathcal{B}'}(\{(T, T')\})$ does not contain pairs of the form $(T_1, \emptyset)$ for every $T_1 \subseteq S \times S$. To prove the case $\leqslant$ of (D.12), use axiom (3.15) with $s_{(T_1,T_2)} := \left(\!\!\left|_{T_1} \mathcal{B}'' \right|\!\!\right)^{T_2}$ for all $T_1, T_2 \subseteq S \times S$, then simply use axioms (3.2), (3.3) and (3.4) appropriately.

To prove the case $\leqslant$ of (D.6), by (D.12), it suffices to show independently that for all $T \subseteq S \times S$, $T' \in 2^{S \times S} \setminus \{\emptyset\}$ and $s, s_2 \in S$,

$$\left(\!\!\left| \mathcal{B}'' \right|\!\!\right)_{T}^{T'} \cdot \mathbf{X}[T', (s_2, s)] \leqslant \left(\sum s' \mid s' \in S : \mathbf{X}[T, (s_2, s')] \cdot \left(\!\!\left| \mathcal{B} \right|\!\!\right)_{s'}^{s}\right) .$$

Let us prove this by case analysis on $T$. For the case $T = \emptyset$, since no block of $\mathcal{B}''$ has $\emptyset$ as starting label (otherwise its ending label would also be $\emptyset$, which is impossible), then, $\left(\!\!\left|_{\emptyset} \mathcal{B}'' \right|\!\!\right)^{T'} = 0$ and so the inequation is proved.

It remains to show the case where $T \in 2^{S\times S}\backslash\{\emptyset\}$. By axiom (3.17), it suffices to show that:

- for all $s'_1, s_4 \in S$ and $[_{T_1}\, m\,]^{T_4} \in (\mathcal{B}'')^1$,

$$m \cdot \mathbf{X}[T_4, (s'_1, s_4)] \leqslant (\sum\, s_1 \mid [\underset{s_1}{\ m\ }]^{s_4} \in \mathcal{B}^1 : \mathbf{X}[(T_1, (s'_1, s_1))] \cdot m) \ , \tag{D.13}$$

- for all $s'_1, s_4 \in S$ and $[_{T_1}\, c{\downarrow_{T_2}}{\uparrow^{T_3}}\, r\,]^{T_4} \in (\mathcal{B}'')^2$,

$$\begin{aligned} & r \cdot \mathbf{X}[T_4, (s'_1, s_4)] \\ & \leqslant \ (\sum\, s_1, s_2, s_3 \mid [\underset{s_1}{\ c\ }{\underset{s_2}{\downarrow}}{\overset{s_3}{\uparrow}}\, r\,]^{s_4} \in \mathcal{B}^2 : \mathbf{X}[T_3, (s_2, s_3)] \cdot \mathbf{X}[T_1, (s'_1, s_1)] \cdot r) \ , \end{aligned} \tag{D.14}$$

- for all $s'_1 \in S$, $[_{T_1}\, c{\downarrow_{T_2}}{\uparrow^{T_3}}\, r\,]^{T_4} \in (\mathcal{B}'')^2$ and $[_{s_1}\, c{\downarrow_{s_2}}{\uparrow^{s_3}}\, r\,]^{s_4} \in \mathcal{B}^2$,

$$\begin{aligned} & (\sum\, s'' \mid s'' \in S : c \cdot \mathbf{X}[T_2, (s_2, s'')] \cdot \underset{s''}{(\!|\, \mathcal{B}\, |\!)}^{s_3} \cdot \mathbf{X}[T_1, (s'_1, s_1)] \cdot r) \\ & \leqslant \ \mathbf{X}[T_1, (s'_1, s_1)] \cdot c \cdot \underset{s_2}{(\!|\, \mathcal{B}\, |\!)}^{s_3} \cdot r \ . \end{aligned} \tag{D.15}$$

For inequation (D.13), first note that

$$m \cdot \mathbf{X}[T_4, (s'_1, s_4)] \leqslant (\sum\, T'_4 \mid [\underset{T_1}{\ m\ }]^{T'_4} \in (\mathcal{B}')^1 : m \cdot \mathbf{X}[T'_4, (s'_1, s_4)]) \ ,$$

by Kleene algebra, by the fact that $(\mathcal{B}'')^1 \Rightarrow (\mathcal{B}')^1$ and by quantification laws. So, inequation (D.13) follows from the previous result and equation (D.10).

We now show (D.14). In the case that $(s'_1, s_4) \notin T_4$, the proof is trivial since $\mathbf{X}[T_4, (s'_1, s_4)] = 0$. So, suppose that $(s'_1, s_4) \in T_4$. By the previous hypothesis, by the definition of $\mathbf{X}$ and by Kleene algebra, it suffices to prove that

$$r \leqslant (\sum\, s_1, s_2, s_3 \mid [\underset{s_1}{\ c\ }{\underset{s_2}{\downarrow}}{\overset{s_3}{\uparrow}}\, r\,]^{s_4} \in \mathcal{B}^2 \wedge (s'_1, s_1) \in T_1 \wedge (s_2, s_3) \in T_3 : r) \ .$$

To prove this, it suffices to prove that there exists $s_1, s_2, s_3 \in S$ such that $[_{s_1}\, c{\downarrow_{s_2}}{\uparrow^{s_3}}\, r\,]^{s_4} \in \mathcal{B}^2$, $(s'_1, s_1) \in T_1$ and $(s_2, s_3) \in T_3$. This is exactly the following result: by definition of $\mathcal{B}''$, condition d, by hypotheses $[_{T_1}\, c{\downarrow_{T_2}}{\uparrow^{T_3}}\, r\,]^{T_4} \in (\mathcal{B}'')^2$ and $(s'_1, s_4) \in T_4$, and set theory, there exists $s_1, s_2, s_3 \in S$ such that $(s'_1, s_1) \in T_1$, $(s_2, s_3) \in T_3$ and $[_{s_1}\, c{\downarrow_{s_2}}{\uparrow^{s_3}}\, r\,]^{s_4} \in \mathcal{B}^2$.

We now show (D.15).

$$
\begin{array}{ll}
 & (\sum s'' \mid s'' \in S : c \cdot \mathbf{X}[T_2, (s_2, s'')] \cdot (\!|_{s''} \mathcal{B} |\!)^{s_3} \cdot \mathbf{X}[T_1, (s_1', s_1)] \cdot r) \\
= & \quad \{\!\{ \text{ Definition of } \mathbf{X} \text{ \& Identity of } \cdot \text{ \& Zero of } \cdot \text{ } \}\!\} \\
 & (\sum s'' \mid s'' \in S \wedge (s_1', s_1) \in T_1 \wedge (s_2, s'') \in T_2 : c \cdot (\!|_{s''} \mathcal{B} |\!)^{s_3} \cdot r) \\
\leqslant & \quad \{\!\{ \text{ Hypothesis } [\!|_{T_1} c \downarrow_{T_2} \uparrow^{T_3} r |\!]^{T_4} \in (\mathcal{B}'')^2 \text{ \& Definition of } \mathcal{B}'', \text{ condition d \&} \\
 & \quad \text{Definition of } \mathsf{Id}. \text{ So, } (s_2, s'') \in T_2 \Rightarrow s'' = s_2 \text{ } \}\!\} \\
 & (\sum s'' \mid s'' \in S \wedge (s_1', s_1) \in T_1 \wedge s'' = s_2 : c \cdot (\!|_{s''} \mathcal{B} |\!)^{s_3} \cdot r) \\
= & \quad \{\!\{ \text{ Definition of } \mathbf{X} \text{ \& Identity of } \cdot \text{ \& Zero of } \cdot \text{ } \}\!\} \\
 & (\sum s'' \mid s'' \in S \wedge s'' = s_2 : \mathbf{X}[T_1, (s_1', s_1)] \cdot c \cdot (\!|_{s''} \mathcal{B} |\!)^{s_3} \cdot r) \\
= & \quad \{\!\{ \text{ One-point rule of } \sum \text{ } \}\!\} \\
 & \mathbf{X}[T_1, (s_1', s_1)] \cdot c \cdot (\!|_{s_2} \mathcal{B} |\!)^{s_3} \cdot r
\end{array}
$$

# Appendix E

# Proof of the Synchronization of Two Deterministic VPA (Theorem 4.9)

Without loss of generality, suppose that $\Gamma_1 \neq \emptyset$ and $\Gamma_2 \neq \emptyset$. The idea of the construction is to use the synchronous product of visibly pushdown automata.

Define the first deterministic visibly pushdown automaton by

$$(S_1 \times S_2, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_1 \times \Gamma_2 \cup \{\bot\}, \delta, I_1 \times I_2, F_1 \times S_2)$$

and the second one by

$$(S_1 \times S_2, \Sigma_{\mathbf{i}}, \Sigma_{\mathbf{c}}, \Sigma_{\mathbf{r}}, \Gamma_1 \times \Gamma_2 \cup \{\bot\}, \delta, I_1 \times I_2, S_1 \times F_2) \ ,$$

where $\delta$ is defined as the set of all the following transitions:

- a transition $((s, s'), a, \lambda; (t, t'), \lambda)$ for all pairs $(s, s') \in S \times S'$, internal actions $a \in \Sigma_{\mathbf{i}}$ and transitions $(s, a, \lambda; t, \lambda) \in \delta_1$ and $(s', a, \lambda; t', \lambda) \in \delta_2$;
- a transition $((s, s'), a, \lambda; (t, t'), (d, d'))$ for all pairs $(s, s') \in S \times S'$, call actions $a \in \Sigma_{\mathbf{c}}$ and transitions $(s, a, \lambda; t, d) \in \delta_1$ and $(s', a, \lambda; t', d') \in \delta_2$;
- a transition $((s, s'), a, (d, d'); (t, t'), \lambda)$ for all pairs $(s, s') \in S \times S'$, return actions $a \in \Sigma_{\mathbf{r}}$, stack symbols $d \in \Gamma_1$, $d' \in \Gamma_2$ and transitions $(s, a, d; t, \lambda) \in \delta_1$ and $(s', a, d'; t', \lambda) \in \delta_2$;
- a transition $((s, s'), a, \bot; (t, t'), \bot)$ for all pairs $(s, s') \in S \times S'$, return actions $a \in \Sigma_{\mathbf{r}}$ and transitions $(s, a, \bot; t, \bot) \in \delta_1$ and $(s', a, \bot; t', \bot) \in \delta_2$.

We only show (4.29), because the proof of (4.30) is similar.

To do this, we use a projection between states of $S_1 \times S_2$ and states of $S_1$. The projection is expressed as a matrix. Let $\mathbf{X}$ be a matrix of size $|S_1 \times S_2| \times |S_1|$ defined for each $s, t \in S_1$ and $s' \in S_2$ by

$$\mathbf{X}[(s, s'), t] := \begin{cases} 1 & \text{if } s = t, \\ 0 & \text{otherwise.} \end{cases}$$

The projection $\mathbf{X}$ is useful since it suffices to prove that

$$\vdash \overrightarrow{I}^t \bullet \mathbf{X} = \overrightarrow{I_1}^t \ , \tag{E.1}$$

$$\vdash \overrightarrow{F_1'} = \mathbf{X} \bullet \overrightarrow{F_1} \ , \tag{E.2}$$

$$\vdash \mathbf{T_c} \bullet \mathbf{X} = \mathbf{X} \bullet \mathbf{T_{c_1}} \ , \tag{E.3}$$

$$\vdash \mathbf{T_\perp} \bullet \mathbf{X} = \mathbf{X} \bullet \mathbf{T_{\perp_1}} \ , \tag{E.4}$$

$$\vdash \mathbf{WM} \bullet \mathbf{X} = \mathbf{X} \bullet \mathbf{WM_1} \ . \tag{E.5}$$

and (4.29) follows easily,

$$\begin{array}{ll} & \overrightarrow{I}^t \bullet (\mathbf{T_\perp} + \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \overrightarrow{F_1'} \\ = & \quad \{\!\{ \text{ Equation (E.2) } \}\!\} \\ & \overrightarrow{I}^t \bullet (\mathbf{T_\perp} + \mathbf{WM})^\star \bullet (\mathbf{T_c} \bullet \mathbf{WM})^\star \bullet \mathbf{X} \bullet \overrightarrow{F_1} \\ = & \quad \{\!\{ \text{ By (E.5) and (E.3), it is direct that } \mathbf{T_c} \bullet \mathbf{WM} \bullet \mathbf{X} = \mathbf{X} \bullet \mathbf{T_{c_1}} \bullet \mathbf{WM_1}. \ \& \\ & \quad \text{Kleene algebra: Bisimulation rule } \}\!\} \\ & \overrightarrow{I}^t \bullet (\mathbf{T_\perp} + \mathbf{WM})^\star \bullet \mathbf{X} \bullet (\mathbf{T_{c_1}} \bullet \mathbf{WM_1})^\star \bullet \overrightarrow{F_1} \\ = & \quad \{\!\{ \text{ By (E.5) and (E.4) and Kleene algebra, it is direct that } (\mathbf{T_\perp} + \mathbf{WM}) \bullet \mathbf{X} = \\ & \quad \mathbf{X} \bullet (\mathbf{T_{\perp_1}} + \mathbf{WM_1}). \ \& \text{ Kleene algebra: Bisimulation rule } \}\!\} \\ & \overrightarrow{I}^t \bullet \mathbf{X} \bullet (\mathbf{T_{\perp_1}} + \mathbf{WM_1})^\star \bullet (\mathbf{T_{c_1}} \bullet \mathbf{WM_1})^\star \bullet \overrightarrow{F_1} \\ = & \quad \{\!\{ \text{ Equation (E.1) } \}\!\} \\ & \overrightarrow{I_1}^t \bullet (\mathbf{T_{\perp_1}} + \mathbf{WM_1})^\star \bullet (\mathbf{T_{c_1}} \bullet \mathbf{WM_1})^\star \bullet \overrightarrow{F_1} \ . \end{array}$$

So, it remains to prove (E.1) to (E.5).

## E.1 Proof of (E.1)

It suffices to show that for all $s \in S_1$,

$$(\overrightarrow{I}^t \bullet \mathbf{X})[1, s] = \overrightarrow{I_1}^t[1, s] \ .$$

$$
\begin{array}{ll}
 & (\overrightarrow{I}^t \bullet \mathbf{X})[1, s] \\
= & \quad \{\!\{ \text{ Definition of } \bullet \ \}\!\} \\
 & (\sum t, t' \mid t \in S_1 \wedge t' \in S_2 : \overrightarrow{I}^t[1, (t, t')] \cdot \mathbf{X}[(t, t'), s]) \\
= & \quad \{\!\{ \text{ Definition of } \mathbf{X} \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } + \text{ \& One-point rule of } \\
 & \qquad \sum \ \}\!\} \\
 & (\sum t' \mid t' \in S_2 : \overrightarrow{I}^t[1, (s, t')]) \\
= & \quad \{\!\{ \text{ Definition of } \overrightarrow{I} \ \}\!\} \\
 & (\sum t' \mid t' \in S_2 \wedge s \in I_1 \wedge t' \in I_2 : 1) \\
= & \quad \{\!\{ \text{ By definition of } S_2 \text{ and } I_2\text{: } I_2 \subseteq S_2 \text{ \& Set theory } \}\!\} \\
 & (\sum t' \mid s \in I_1 \wedge t' \in I_2 : 1) \\
= & \quad \{\!\{ \text{ Definition of } \overrightarrow{I_1} \ \}\!\} \\
 & (\sum t' \mid t' \in I_2 : \overrightarrow{I_1}^t[1, s]) \\
= & \quad \{\!\{ \text{ Hypothesis: The automaton } \mathcal{A}_2 \text{ is deterministic \& Definition of deter-} \\
 & \qquad \text{minism: there is one and only one initial state } \}\!\} \\
 & \overrightarrow{I_1}^t[1, s]
\end{array}
$$

## E.2 Proof of (E.2)

It suffices to show that for all $s \in S_1$ and $s' \in S_2$,

$$\overrightarrow{F_1'}[(s, s'), 1] = (\mathbf{X} \bullet \overrightarrow{F_1})[(s, s'), 1] \ .$$

$$
\begin{array}{ll}
 & (\mathbf{X} \bullet \overrightarrow{F_1})[(s, s'), 1] \\
= & \quad \{\!\{ \text{ Definition of } \bullet \ \}\!\} \\
 & (\sum t \mid t \in S_1 : \mathbf{X}[(s, s'), t] \cdot \overrightarrow{F_1}[t, 1]) \\
= & \quad \{\!\{ \text{ Definition of } \mathbf{X} \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } + \text{ \& One-point rule of } \\
 & \qquad \sum \ \}\!\} \\
 & \overrightarrow{F_1}[s, 1] \\
= & \quad \{\!\{ \text{ Definition of } \overrightarrow{F_1'} \ \}\!\} \\
 & \overrightarrow{F_1'}[(s, s'), 1]
\end{array}
$$

## E.3 Proof of (E.3)

It suffices to show that for all $s, t \in S_1$ and $s' \in S_2$,

$$(\mathbf{T}_{\mathsf{c}} \bullet \mathbf{X})[(s,s'),t] = (\mathbf{X} \bullet \mathbf{T}_{\mathsf{c}_1})[(s,s'),t] \ .$$

First note that, by Definition 4.3 of $\mathbf{T}_{\mathsf{c}}$ and $\mathbf{T}_{\mathsf{c}_1}$, these two matrices can be expressed by the following summations:

$$\begin{aligned}
\mathbf{T}_{\mathsf{c}_1} &= (\sum\, a, d \mid a \in \Sigma_{\mathbf{c}} \wedge d \in \Gamma_1 : \mathbf{T}_{\mathsf{c}_1}^{a,d}) \ , \\
\mathbf{T}_{\mathsf{c}} &= (\sum\, a, d, d' \mid a \in \Sigma_{\mathbf{c}} \wedge d \in \Gamma_1 \wedge d' \in \Gamma_2 : \mathbf{T}_{\mathsf{c}}^{a,(d,d')}) \ ,
\end{aligned}$$

where each matrix $\mathbf{T}_{\mathsf{c}_1}^{a,d}$ is of size $|S_1| \times |S_1|$ and contains only entries of the form 0 or $a$ and each matrix $\mathbf{T}_{\mathsf{c}}^{a,(d,d')}$ is of size $|S_1 \times S_2| \times |S_1 \times S_2|$ and contains only entries of the form 0 or $a$. These matrices are a particular form of simple matrices as used by Kozen in [25]. Note also that, by definition of the construction, for all $s, t \in S_1$, $s', t' \in S_2$, $a \in \Sigma_{\mathbf{c}}$, $d \in \Gamma_1$ and $d' \in \Gamma_2$,

$$\mathbf{T}_{\mathsf{c}}^{a,(d,d')}[(s,s'),(t,t')] = \begin{cases} a & \text{if } (s,a,\lambda;t,d) \in \delta_1 \text{ and } (s',a,\lambda;t',d') \in \delta_2, \\ 0 & \text{otherwise.} \end{cases}$$

In other words, for all $s, t \in S_1$, $s', t' \in S_2$, $a \in \Sigma_{\mathbf{c}}$, $d \in \Gamma_1$ and $d' \in \Gamma_2$,

$$\mathbf{T}_{\mathsf{c}}^{a,(d,d')}[(s,s'),(t,t')] = \begin{cases} a & \text{if } \mathbf{T}_{\mathsf{c}_1}^{a,d}[s,t] = a \text{ and } (s',a,\lambda;t',d') \in \delta_2, \\ 0 & \text{otherwise.} \end{cases}$$

Using these definitions, the proof is possible.

$$\begin{array}{ll}
 & (\mathbf{T}_{\mathsf{c}} \bullet \mathbf{X})[(s,s'),t] \\
= & \quad \{\!\{ \text{ Definition of } \bullet \ \}\!\} \\
 & (\sum\, u, u' \mid u \in S_1 \wedge u' \in S_2 : \mathbf{T}_{\mathsf{c}}[(s,s'),(u,u')] \cdot \mathbf{X}[(u,u'),t]) \\
= & \quad \{\!\{ \text{ Definition of } \mathbf{X} \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } + \text{ \& One-point rule of } \sum \ \}\!\} \\
 & (\sum\, u' \mid u' \in S_2 : \mathbf{T}_{\mathsf{c}}[(s,s'),(t,u')]) \\
= & \quad \{\!\{ \text{ Definition of } \mathbf{T}_{\mathsf{c}} \ \}\!\} \\
 & (\sum\, u' \mid u' \in S_2 : (\sum\, a, d, d' \mid a \in \Sigma_{\mathbf{c}} \wedge d \in \Gamma_1 \wedge d' \in \Gamma_2 : \mathbf{T}_{\mathsf{c}}^{a,(d,d')})[(s,s'),(t,u')]) \\
= & \quad \{\!\{ \text{ Definition of } + \text{ \& Quantification laws } \}\!\} \\
 & (\sum\, a, d \mid a \in \Sigma_{\mathbf{c}} \wedge d \in \Gamma_1 : (\sum\, u', d' \mid u' \in S_2 \wedge d' \in \Gamma_2 : \mathbf{T}_{\mathsf{c}}^{a,(d,d')}[(s,s'),(t,u')])) \\
= & \quad \{\!\{ \text{ Hypothesis: The automaton } \mathcal{A}_2 \text{ is deterministic \& Definition of determinism: there is one and only one stack symbol } d' \in \Gamma_2 \text{ and state } u' \text{ reachable from } s' \text{ by reading } a \in \Sigma_{\mathbf{c}} \text{ such that } (s',a,\lambda;u',d') \in \delta_2 \text{ \& Definition of the construction } \}\!\}
\end{array}$$

$$
\begin{array}{ll}
 & (\sum a, d \mid a \in \Sigma_{\mathbf{c}} \wedge d \in \Gamma_1 : \mathbf{T}_{\mathsf{c}_1}^{a,d}[s,t]) \\
= & \quad \{\!\!\{ \text{ Definition of } + \ \}\!\!\} \\
 & (\sum a, d \mid a \in \Sigma_{\mathbf{c}} \wedge d \in \Gamma_1 : \mathbf{T}_{\mathsf{c}_1}^{a,d})[s,t] \\
= & \quad \{\!\!\{ \text{ Definition of } \mathbf{T}_{\mathsf{c}_1} \ \}\!\!\} \\
 & \mathbf{T}_{\mathsf{c}_1}[s,t] \\
= & \quad \{\!\!\{ \text{ One-point rule of } \sum \ \}\!\!\} \\
 & (\sum u \mid u \in S_1 \wedge s = u : \mathbf{T}_{\mathsf{c}_1}[u,t]) \\
= & \quad \{\!\!\{ \text{ Definition of } \mathbf{X} \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } + \ \}\!\!\} \\
 & (\sum u \mid u \in S_1 : \mathbf{X}[(s,s'),u] \cdot \mathbf{T}_{\mathsf{c}_1}[u,t]) \\
= & \quad \{\!\!\{ \text{ Definition of } \bullet \ \}\!\!\} \\
 & (\mathbf{X} \bullet \mathbf{T}_{\mathsf{c}_1})[(s,s'),t]
\end{array}
$$

## E.4 Proof of (E.4)

It suffices to show that for all $s, t \in S_1$ and $s' \in S_2$,

$$(\mathbf{T}_{\perp} \bullet \mathbf{X})[(s,s'),t] = (\mathbf{X} \bullet \mathbf{T}_{\perp_1})[(s,s'),t] \ .$$

First note that, by Definition 4.3 of $\mathbf{T}_{\perp}$ and $\mathbf{T}_{\perp_1}$, these two matrices can be expressed by the following summations:

$$
\begin{aligned}
\mathbf{T}_{\perp_1} &= (\sum a \mid a \in \Sigma_{\mathbf{r}} : \mathbf{T}_{\perp_1}^{a}) \ , \\
\mathbf{T}_{\perp} &= (\sum a \mid a \in \Sigma_{\mathbf{r}} : \mathbf{T}_{\perp}^{a}) \ ,
\end{aligned}
$$

where each matrix $\mathbf{T}_{\perp_1}^{a}$ is of size $|S_1| \times |S_1|$ and contains only entries of the form 0 or $a$ and each matrix $\mathbf{T}_{\perp}^{a}$ is of size $|S_1 \times S_2| \times |S_1 \times S_2|$ and contains only entries of the form 0 or $a$. These matrices are a particular form of simple matrices as used by Kozen in [25]. Note also that, by definition of the construction, for all $s, t \in S_1$, $s', t' \in S_2$ and $a \in \Sigma_{\mathbf{r}}$,

$$\mathbf{T}_{\perp}^{a}[(s,s'),(t,t')] = \begin{cases} a & \text{if } (s,a,\perp;t,\perp) \in \delta_1 \text{ and } (s',a,\perp;t',\perp) \in \delta_2, \\ 0 & \text{otherwise.} \end{cases}$$

In other words, for all $s, t \in S_1$, $s', t' \in S_2$ and $a \in \Sigma_{\mathbf{r}}$,

$$\mathbf{T}_{\perp}^{a}[(s,s'),(t,t')] = \begin{cases} a & \text{if } \mathbf{T}_{\perp_1}^{a}[s,t] = a \text{ and } (s',a,\perp;t',\perp) \in \delta_2, \\ 0 & \text{otherwise.} \end{cases}$$

Using these definitions, the proof is possible.

$$
\begin{array}{ll}
 & (\mathbf{T}_{\perp} \bullet \mathbf{X})[(s,s'),t] \\
= & \quad \{\!\{ \text{ Definition of } \bullet \ \}\!\} \\
 & (\sum u,u' \mid u \in S_1 \wedge u' \in S_2 : \mathbf{T}_{\perp}[(s,s'),(u,u')] \cdot \mathbf{X}[(u,u'),t]) \\
= & \quad \{\!\{ \text{ Definition of } \mathbf{X} \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } + \text{ \& One-point rule of } \sum \ \}\!\} \\
 & (\sum u' \mid u' \in S_2 : \mathbf{T}_{\perp}[(s,s'),(t,u')]) \\
= & \quad \{\!\{ \text{ Definition of } \mathbf{T}_{\perp} \ \}\!\} \\
 & (\sum u' \mid u' \in S_2 : (\sum a \mid a \in \Sigma_{\mathbf{r}} : \mathbf{T}^{a}_{\perp})[(s,s'),(t,u')]) \\
= & \quad \{\!\{ \text{ Definition of } + \text{ \& Quantification laws } \}\!\} \\
 & (\sum a \mid a \in \Sigma_{\mathbf{r}} : (\sum u' \mid u' \in S_2 : \mathbf{T}^{a}_{\perp}[(s,s'),(t,u')])) \\
= & \quad \{\!\{ \text{ Hypothesis: The automaton } \mathcal{A}_2 \text{ is deterministic \& Definition of determinism: there is one and only one state } u' \text{ reachable from } s' \text{ by reading } a \in \Sigma_{\mathbf{r}} \text{ such that } (s',a,\perp;u',\perp) \in \delta_2 \text{ \& Definition of the construction } \}\!\} \\
 & (\sum a \mid a \in \Sigma_{\mathbf{r}} : \mathbf{T}^{a}_{\perp_1}[s,t]) \\
= & \quad \{\!\{ \text{ Definition of } + \ \}\!\} \\
 & (\sum a \mid a \in \Sigma_{\mathbf{r}} : \mathbf{T}^{a}_{\perp_1})[s,t] \\
= & \quad \{\!\{ \text{ Definition of } \mathbf{T}_{\perp_1} \ \}\!\} \\
 & \mathbf{T}_{\perp_1}[s,t] \\
= & \quad \{\!\{ \text{ One-point rule of } \sum \ \}\!\} \\
 & (\sum u \mid u \in S_1 \wedge s = u : \mathbf{T}_{\perp_1}[u,t]) \\
= & \quad \{\!\{ \text{ Definition of } \mathbf{X} \text{ \& Zero of } \cdot \text{ \& Identity of } \cdot \text{ and } + \ \}\!\} \\
 & (\sum u \mid u \in S_1 : \mathbf{X}[(s,s'),u] \cdot \mathbf{T}_{\perp_1}[u,t]) \\
= & \quad \{\!\{ \text{ Definition of } \bullet \ \}\!\} \\
 & (\mathbf{X} \bullet \mathbf{T}_{\perp_1})[(s,s'),t]
\end{array}
$$

## E.5 Proof of (E.5)

It suffices to show that for all $s,t \in S_1$ and $s' \in S_2$,

$$(\mathbf{WM} \bullet \mathbf{X})[(s,s'),t] = (\mathbf{X} \bullet \mathbf{WM}_1)[(s,s'),t] \ .$$

By definition of $\bullet$, $\mathbf{WM}$ and $\mathbf{WM}_1$, it suffices to show that for all $s,t \in S_1$ and $s' \in S_2$,

$$
\begin{array}{l}
(\sum u,u' \mid u \in S_1 \wedge u' \in S_2 : \ (\!|\underset{(s,s')}{\ } \mathcal{B} \overset{(u,u')}{\ }|\!) \cdot \mathbf{X}[(u,u'),t]) \\
= \ (\sum u \mid u \in S_1 : \mathbf{X}[(s,s'),u] \cdot (\!|\underset{u}{\ } \mathcal{B}_1 \overset{t}{\ }|\!)) \ .
\end{array}
\tag{E.6}
$$

We first explicitly describe the list of blocks of $\mathcal{B}$ with respect to $\mathcal{B}_1$ and $\mathcal{B}_2$.

a. The set of labels $\{(s_i, s'_j) \mid s_i \in S_1 \wedge s'_j \in S_2\}$ is defined.

b. For all $s_i \in S_1$ and $s'_j \in S_2$,

$$\underset{(s_i,s'_j)}{[} \; 1 \; \overset{(s_i,s'_j)}{]} \in \mathcal{B}^1 \enspace .$$

c. For all internal actions $a \in \Sigma_{\mathbf{i}}$, $s_i, s_k \in S_1$ and $s'_j, s'_l \in S_2$,

$$\underset{(s_i,s'_j)}{[} \; a \; \overset{(s_k,s'_l)}{]} \in \mathcal{B}^1 \quad \Leftrightarrow \quad \underset{s_i}{[} \; a \; \overset{s_k}{]} \in \mathcal{B}_1^1 \quad \wedge \quad \underset{s'_j}{[} \; a \; \overset{s'_l}{]} \in \mathcal{B}_2^1 \enspace .$$

d. For all call actions $c \in \Sigma_{\mathbf{c}}$, return actions $r \in \Sigma_{\mathbf{r}}$, $s_i, s_k, s_m, s_o \in S_1$ and $s'_j, s'_l, s'_n, s'_p \in S_2$,

$$\underset{(s_i,s'_j)}{[} \; c \; \underset{(s_m,s'_n)}{\downarrow} \; \overset{(s_o,s'_p)}{\uparrow} \; r \; \overset{(s_k,s'_l)}{]} \in \mathcal{B}^2 \quad \Leftrightarrow \quad \underset{s_i}{[} \; c \underset{s_m}{\downarrow} \overset{s_o}{\uparrow} r \; \overset{s_k}{]} \in \mathcal{B}_1^2 \quad \wedge \quad \underset{s'_j}{[} \; c \underset{s'_n}{\downarrow} \overset{s'_p}{\uparrow} r \; \overset{s'_l}{]} \in \mathcal{B}_2^2 \enspace .$$

We now show the case $\leqslant$ of (E.6). By the definition of $\mathbf{X}$, by Kleene algebra and by quantification laws, it suffices to prove that for all $s, t \in S_1$ and $s' \in S_2$,

$$(\sum u' \mid u' \in S_2 : \underset{(s,s')}{(\!|} \; \mathcal{B} \; \overset{(t,u')}{|\!)}) \leqslant \underset{s}{(\!|} \mathcal{B}_1 \overset{t}{|\!)} \enspace . \tag{E.7}$$

By Kleene algebra, it suffices to prove independently that for all $s, t \in S_1$ and $s', u' \in S_2$,

$$\underset{(s,s')}{(\!|} \; \mathcal{B} \; \overset{(t,u')}{|\!)} \leqslant \underset{s}{(\!|} \mathcal{B}_1 \overset{t}{|\!)} \enspace .$$

Using (3.37) with the projection function $f : S_1 \times S_2 \to S_1$ defined, for all $v \in S_1$ and $v' \in S_2$, by

$$f((v, v')) := v \enspace ,$$

it can be proved that

$$\underset{(s,s')}{(\!|} \; \mathcal{B} \; \overset{(t,u')}{|\!)} \quad \leqslant \quad \underset{f((s,s'))}{(\!|} \; \widehat{f}(\mathcal{B}) \; \overset{f((t,u'))}{|\!)} \quad = \quad \underset{s}{(\!|} \widehat{f}(\mathcal{B}) \overset{t}{|\!)} \enspace .$$

Note now that $\widehat{f}(\mathcal{B})$ is a sublist of $\mathcal{B}_1$. This is direct from the definition of $\mathcal{B}$, conditions a to d. So, the result follows from (3.26) and (3.30).

We now show the case $\geqslant$ of (E.6). By Kleene algebra, it suffices to show independently that, for all $s, t, u \in S_1$ and $s' \in S_2$,

$$\mathbf{X}[(s, s'), u] \cdot \underset{u}{(\!|} \mathcal{B}_1 \overset{t}{|\!)} \leqslant (\sum u', u'' \mid u' \in S_1 \wedge u'' \in S_2 : \underset{(s,s')}{(\!|} \; \mathcal{B} \; \overset{(u',u'')}{|\!)} \cdot \mathbf{X}[(u', u''), t]) \enspace .$$

By (3.16) using $b_{v'}^{(v_2,v)} := \mathbf{X}[v', v]$ for all $v_2, v \in S_1$ and $v' \in S_1 \times S_2$, it suffices to prove independently that

- for all $(t_1, u_1) \in S_1 \times S_2$ and $[_{s_1}\, m\, ]^{s_4} \in \mathcal{B}_1^1$,

$$\mathbf{X}[((t_1,u_1),s_1)] \cdot m \leqslant (\sum t_4, u_4 \mid [\underset{(t_1,u_1)}{\ } m \overset{(t_4,u_4)}{\ }] \in \mathcal{B}^1 : m \cdot \mathbf{X}[((t_4,u_4),s_4)]) \ , \quad \text{(E.8)}$$

- for all $(t_1, u_1) \in S_1 \times S_2$ and $[_{s_1}\, c{\downarrow_{s_2}}{\uparrow^{s_3}}\, r\, ]^{s_4} \in \mathcal{B}_1^2$,

$$\begin{aligned} &\mathbf{X}[(t_1,u_1),s_1] \cdot c \\ &\leqslant (\sum t_2,u_2,t_3,u_3,t_4,u_4 \mid [\underset{(t_1,u_1)}{\ } c \underset{(t_2,u_2)}{\downarrow} \overset{(t_3,u_3)}{\uparrow} r \overset{(t_4,u_4)}{\ }] \in \mathcal{B}^2 : \\ &\qquad \mathbf{X}[(t_1,u_1),s_1] \cdot c \cdot \mathbf{X}[(t_2,u_2),s_2] \cdot \mathbf{X}[(t_2,u_2),s_2]) \ , \end{aligned} \quad \text{(E.9)}$$

- for all $(t_1, u_1), (t_2, u_2), (t_3', u_3') \in S_1 \times S_2$ and $[_{s_1}\, c{\downarrow_{s_2}}{\uparrow^{s_3}}\, r\, ]^{s_4} \in \mathcal{B}_1^2$,

$$\begin{aligned} &(\sum t_3,u_3,t_4,u_4 \mid [\underset{(t_1,u_1)}{\ } c \underset{(t_2,u_2)}{\downarrow} \overset{(t_3,u_3)}{\uparrow} r \overset{(t_4,u_4)}{\ }] \in \mathcal{B}^2 : \\ &\qquad \mathbf{X}[(t_1,u_1),s_1] \cdot c \cdot \mathbf{X}[(t_2,u_2),s_2] \cdot (\!|\ \mathcal{B}\ |\!)_{(t_2,u_2)}^{(t_3',u_3')} \cdot \mathbf{X}[(t_3',u_3'),s_3] \cdot r) \\ &\leqslant (\sum t_4,u_4 \mid [\underset{(t_1,u_1)}{\ } c \underset{(t_2,u_2)}{\downarrow} \overset{(t_3',u_3')}{\uparrow} r \overset{(t_4,u_4)}{\ }] \in \mathcal{B}^2 : \\ &\qquad c \cdot (\!|\ \mathcal{B}\ |\!)_{(t_2,u_2)}^{(t_3',u_3')} \cdot r \cdot \mathbf{X}[(t_4,u_4),s_4]) \ . \end{aligned} \quad \text{(E.10)}$$

We first show (E.8). For the case $s_1 \neq t_1$, then $\mathbf{X}[(t_1, u_1), s_1] = 0$ and so the inequation is trivial since 0 is the minimum of the algebra. For the case $s_1 = t_1$, then $\mathbf{X}[(t_1, u_1), s_1] = 1$. Let us prove this by case analysis on $m$. The case $m = 0$ is trivial by Kleene algebra.

For the case $m = 1$, by condition B of the definition of $\mathcal{B}_1$, $s_1 = s_4$. Also, by condition b of the definition of $\mathcal{B}$, $(t_4, u_4) = (t_1, u_1)$. So, using the one-point rule of $\sum$ for the right-hand side of the inequation, we have to show that

$$\mathbf{X}[((t_1,u_1),s_1)] \cdot 1 \leqslant 1 \cdot \mathbf{X}[((t_1,u_1),s_1)] \ .$$

This is trivial by the identity of $\cdot$.

For the case $m \in \Sigma_{\mathbf{i}}$, by the definition of $\mathbf{X}$, Kleene algebra and the one-point rule of $\sum$, we have to prove that

$$m \leqslant (\sum u_4 \mid \underset{(s_1,u_1)}{[} m \overset{(s_4,u_4)}{]} \in \mathcal{B}^1 : m) \ .$$

By Kleene algebra, it suffices to prove that there exists $u_4 \in S_2$ such that $[_{(s_1,u_1)} m ]^{(s_4,u_4)} \in \mathcal{B}^1$. By condition c of the definition of $\mathcal{B}$, it suffices to prove that there exists a $u_4 \in S_2$ such that $[_{s_1} m ]^{s_4} \in \mathcal{B}_1^1$ and $[_{u_1} m ]^{u_4} \in \mathcal{B}_2^1$. By the hypothesis stating that $[_{s_1} m ]^{s_4} \in \mathcal{B}_1^1$ and by the definition of $\mathcal{B}_2$, it suffices to prove that there exists a $u_4 \in S_2$ such that $(u_1, m, \lambda; u_4, \lambda) \in \delta_2$. This is always the case since the automaton $\mathcal{A}_2$ is deterministic by hypothesis. In other words, by the definition of determinism, for every $u_1 \in S_2$ and $m \in \Sigma_{\mathbf{i}}$, there exists $u_4 \in S_2$ reachable from $u_1$ by reading $m$, that is to say $(u_1, m, \lambda; u_4, \lambda) \in \delta_2$.

We now show (E.9). For the case $s_1 \neq t_1$, then $\mathbf{X}[(t_1, u_1), s_1] = 0$ and so the inequation is trivial since 0 is the minimum of the algebra. For the case $s_1 = t_1$, then $\mathbf{X}[(t_1, u_1), s_1] = 1$. By the definition of $\mathbf{X}$, Kleene algebra and the one-point rule of $\sum$, we have to prove that

$$c \leqslant (\sum u_2, t_3, u_3, t_4, u_4 \mid \underset{(s_1,u_1)}{[} c \underset{(s_2,u_2)}{\downarrow} \overset{(t_3,u_3)}{\uparrow} r \overset{(t_4,u_4)}{]} \in \mathcal{B}^2 : c) \ .$$

By Kleene algebra, it suffices to prove that there exists $t_3, t_4 \in S_1$ and $u_2, u_3, u_4 \in S_2$ such that $[_{(s_1,u_1)} c{\downarrow_{(s_2,u_2)}} {\uparrow^{(t_3,u_3)}} r ]^{(t_4,u_4)} \in \mathcal{B}^2$. By condition d of the definition of $\mathcal{B}$ and quantification laws, it suffices to prove that there exists $t_3, t_4 \in S_1$ such that $[_{s_1} c{\downarrow_{s_2}} {\uparrow^{t_3}} r ]^{t_4} \in \mathcal{B}_1^2$ and there exists $u_2, u_3, u_4 \in S_2$ such that $[_{u_1} c{\downarrow_{u_2}} {\uparrow^{u_3}} r ]^{u_4} \in \mathcal{B}_2^2$. By hypothesis, $[_{s_1} c{\downarrow_{s_2}} {\uparrow^{s_3}} r ]^{s_4} \in \mathcal{B}_1^2$, so there exists at least one $t_3, t_4 \in S_1$ such that $[_{s_1} c{\downarrow_{s_2}} {\uparrow^{t_3}} r ]^{t_4} \in \mathcal{B}_1^2$. It remains to prove that there exists $u_2, u_3, u_4 \in S_2$ such that $[_{u_1} c{\downarrow_{u_2}} {\uparrow^{u_3}} r ]^{u_4} \in \mathcal{B}_2^2$. By the definition of $\mathcal{B}_2$, it suffices to prove that there exists $u_2, u_3, u_4 \in S_2$ and $d' \in \Gamma_2$ such that $(u_1, c, \lambda; u_2, d') \in \delta_2$ and $(u_3, r, d'; u_4, \lambda) \in \delta_2$. By the hypothesis that the automaton $\mathcal{A}_2$ is deterministic and the definition of determinism, for every $u_1 \in S_2$ and $c \in \Sigma_{\mathbf{c}}$, there exists $d' \in \Gamma_2$ and a state $u_2 \in S_2$ reachable from $u_1$ by reading $c$ such that $(u_1, c, \lambda; u_2, d') \in \delta_2$ and for every $u_3 \in S_2$, $r \in \Sigma_{\mathbf{r}}$ and $d' \in \Gamma_2$, there exists a state $u_4 \in S_2$ reachable from $u_3$ by reading $r$ when having $d'$ as top symbol of the stack, that is to say, $(u_3, r, d'; u_4, \lambda) \in \delta_2$. Since $S_2 \neq \emptyset$ by hypothesis, there is at least one $u_3 \in S_2$.

We now show (E.10). By Kleene algebra, the definition of $\mathbf{X}$ and the one-point rule of $\sum$, it suffices to prove that for all $[_{(s_1,u_1)} c{\downarrow_{(s_2,u_2)}} {\uparrow^{(t_3,u_3)}} r ]^{(t_4,u_4)} \in \mathcal{B}^2$,

$$c \cdot \underset{(s_2,u_2)}{(\!|} \mathcal{B} \overset{(s_3,u_3')}{|\!)} \cdot r \leqslant (\sum u_4' \mid \underset{(s_1,u_1)}{[} c \underset{(s_2,u_2)}{\downarrow} \overset{(s_3,u_3')}{\uparrow} r \overset{(s_4,u_4')}{]} \in \mathcal{B}^2 : c \cdot \underset{(s_2,u_2)}{(\!|} \mathcal{B} \overset{(s_3,u_3')}{|\!)} \cdot r) \ .$$

By Kleene algebra, it suffices to prove that there exists a $u'_4 \in S_2$ such that the binary block $\lfloor_{(s_1,u_1)} c \downarrow_{(s_2,u_2)} \uparrow^{(s_3,u'_3)} r \rfloor^{(s_4,u'_4)}$ is in $\mathcal{B}^2$. By condition d of the definition of $\mathcal{B}$, it suffices to prove that there exists a $u'_4 \in S_2$ such that $\lfloor_{s_1} c \downarrow_{s_2} \uparrow^{s_3} r \rfloor^{s_4} \in \mathcal{B}_1^2$ and $\lfloor_{u_1} c \downarrow_{u_2} \uparrow^{u'_3} r \rfloor^{u'_4} \in \mathcal{B}_2^2$. By hypothesis, $\lfloor_{s_1} c \downarrow_{s_2} \uparrow^{s_3} r \rfloor^{s_4} \in \mathcal{B}_1^2$, so it remains to prove that there exists a $u'_4 \in S_2$ such that $\lfloor_{u_1} c \downarrow_{u_2} \uparrow^{u'_3} r \rfloor^{u'_4} \in \mathcal{B}_2^2$. By the definition of $\mathcal{B}_2$, it suffices to prove that there exists a $u'_4 \in S_2$ and a $d' \in \Gamma_2$ such that $(u_1, c, \lambda; u_2, d') \in \delta_2$ and $(u'_3, r, d'; u'_4, \lambda) \in \delta_2$. By the hypothesis $\lfloor_{(s_1,u_1)} c \downarrow_{(s_2,u_2)} \uparrow^{(t_3,u_3)} r \rfloor^{(t_4,u_4)} \in \mathcal{B}^2$, by condition d of the definition of $\mathcal{B}$ and by the definition of $\mathcal{B}_2$, there exists a $d'' \in \Gamma_2$ such that $(u_1, c, \lambda; u_2, d'') \in \delta_2$. So, it suffices to show that for this $d''$, there exists a $u'_4 \in S_2$ such that $(u'_3, r, d''; u'_4, \lambda) \in \delta_2$. This is immediate from the hypothesis that the automaton $\mathcal{A}_2$ is deterministic. In other words, by the definition of determinism, for every $u'_3 \in S_2$, $r \in \Sigma_{\mathbf{r}}$ and $d'' \in \Gamma_2$, there exists a state $u'_4 \in S_2$ reachable from $u'_3$ by reading $r$ when having $d''$ as top symbol of the stack, that is to say, $(u'_3, r, d''; u'_4, \lambda) \in \delta_2$.

# Appendix F

# Proofs of Three Results on the Function $\mathsf{mb\_vpre\_suffixes}_\theta$ (Lemmas 5.5 to 5.7)

We prove that Lemmas 5.5 to 5.7 are valid.

## F.1 Proof of Lemma 5.5

The proof is done by structural induction on $[_x\, m\,]^y$. For the base case where $[_x\, m\,]^y$ is such that $m \in \Sigma_{\mathbf{i}} \cup \mathsf{Tests}_{\mathbf{B}}$, then, by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$ and $\mathsf{mb\_vpre}_\theta$,

$$(\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, m\, \overset{y}{]}))(x) = m = \mathsf{mb\_vpre}_\theta(m) \ .$$

For the base case where $[_x\, m\,]^y$ is a metablock $[_x(c{\downarrow_z}{\uparrow^w}\, r)\,]^y$, by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$ and $\mathsf{mb\_vpre}_\theta$,

$$(\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}(c\underset{z}{\downarrow}\overset{w}{\uparrow} r)\overset{y}{]}))(x) = c \cdot \theta(z,w) \cdot r = \mathsf{mb\_vpre}_\theta((c\underset{z}{\downarrow}\overset{w}{\uparrow} r)) \ .$$

For the inductive case where $[_x\, m\,]^y$ is a metablock $[_x\, p \cdot q\,]^y$, we suppose that $z$ is

the fresh label used by $\mathsf{mb\_vpre\_suffixes}_\theta$. So, we suppose that

$$(\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \overset{z}{]}))(x) = \mathsf{mb\_vpre}_\theta(p) \ , \tag{F.1}$$

$$(\mathsf{mb\_vpre\_suffixes}_\theta(\underset{z}{[}\, q \overset{y}{]}))(z) = \mathsf{mb\_vpre}_\theta(q) \ , \tag{F.2}$$

and we must prove that

$$(\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \cdot q \overset{y}{]}))(x) = \mathsf{mb\_vpre}_\theta(p \cdot q) \ .$$

Note that inductive hypothesis (F.1) states that

$$(x \mapsto \mathsf{mb\_vpre}_\theta(p)) \in \mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \overset{z}{]}) \ .$$

By the previous reasoning and the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$, we have that

$$(\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \cdot q \overset{y}{]}))(x) = \mathsf{mb\_vpre}_\theta(p) \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{z}{[}\, q \overset{y}{]}))(z) \ .$$

Using inductive hypothesis (F.2), we have that

$$(\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \cdot q \overset{y}{]}))(x) = \mathsf{mb\_vpre}_\theta(p) \cdot \mathsf{mb\_vpre}_\theta(q) \ .$$

So, the result follows from the definition of $\mathsf{mb\_vpre}_\theta$ for the case $\cdot$.

For the inductive case where $[_x\, m\, ]^y$ is a metablock $[_x\, p + q\, ]^y$, we suppose that

$$(\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \overset{y}{]}))(x) = \mathsf{mb\_vpre}_\theta(p) \ , \tag{F.3}$$

$$(\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, q \overset{y}{]}))(x) = \mathsf{mb\_vpre}_\theta(q) \ , \tag{F.4}$$

and we must prove that

$$(\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p + q \overset{y}{]}))(x) = \mathsf{mb\_vpre}_\theta(p + q) \ .$$

We are able to prove it.

$$\begin{array}{ll} & (\mathsf{mb\_vpre\_suffixes}_\theta([_x\, p + q\, ]^y))(x) \\ = & \qquad \{\!\{ \text{ Definition of } \mathsf{mb\_vpre\_suffixes}_\theta \ \}\!\} \\ & (\mathsf{mb\_vpre\_suffixes}_\theta([_x\, p\, ]^y))(x) + (\mathsf{mb\_vpre\_suffixes}_\theta([_x\, q\, ]^y))(x) \\ = & \qquad \{\!\{ \text{ Induction hypotheses (F.3) and (F.4) } \}\!\} \\ & \mathsf{mb\_vpre}_\theta(p) + \mathsf{mb\_vpre}_\theta(q) \\ = & \qquad \{\!\{ \text{ Definition of } \mathsf{mb\_vpre}_\theta \ \}\!\} \\ & \mathsf{mb\_vpre}_\theta(p + q) \end{array}$$

For the inductive case where $[\!]_x\, m\, [\!]^y$ is a metablock $[\!]_x\, p^*\, [\!]^y$, we suppose that

$$(\mathsf{mb\_vpre\_suffixes}_\theta([\!]_x\, p\, [\!]^y))(x) = \mathsf{mb\_vpre}_\theta(p) \ , \tag{F.5}$$

and we must prove that

$$(\mathsf{mb\_vpre\_suffixes}_\theta([\!]_x\, p^*\, [\!]^y))(x) = \mathsf{mb\_vpre}_\theta(p^*) \ .$$

We are able to prove it.

$$\begin{array}{ll} & (\mathsf{mb\_vpre\_suffixes}_\theta([\!]_x\, p^*\, [\!]^y))(x) \\ = & \qquad \{\!\!\{ \text{ Definition of } \mathsf{mb\_vpre\_suffixes}_\theta \ \}\!\!\} \\ & ((\mathsf{mb\_vpre\_suffixes}_\theta([\!]_x\, p\, [\!]^y))(x))^* \\ = & \qquad \{\!\!\{ \text{ Induction hypothesis (F.5) } \}\!\!\} \\ & (\mathsf{mb\_vpre}_\theta(p))^* \\ = & \qquad \{\!\!\{ \text{ Definition of } \mathsf{mb\_vpre}_\theta \ \}\!\!\} \\ & \mathsf{mb\_vpre}_\theta(p^*) \end{array}$$

## F.2 Proof of Lemma 5.6

The proof is done by structural induction on $[\!]_x\, m\, [\!]^y$. For the base case where $[\!]_x\, m\, [\!]^y$ is such that $m \in \Sigma_{\mathbf{i}} \cup \mathsf{Tests}_{\mathbf{B}}$, then $\mathsf{mb}([\!]_x\, m\, [\!]^y)^1 = \{[\!]_x\, m\, [\!]^y\}$ and $\mathsf{mb}([\!]_x\, m\, [\!]^y)^2 = \emptyset$. So, we have to prove that

$$m \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta([\!]_x\, m\, [\!]^y))(x) \ .$$

This is trivial by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$.

For the base case where $[\!]_x\, m\, [\!]^y$ is a metablock $[\!]_x (c{\downarrow_u}{\uparrow^{u'}} r)\, [\!]^y$, then we have

$$\mathsf{mb}([\!]_x\, m\, [\!]^y)^1 = \emptyset \qquad \text{and} \qquad \mathsf{mb}([\!]_x\, m\, [\!]^y)^2 = \{[\!]_x\, c {\downarrow_u}{\uparrow^{u'}} r\, [\!]^y\} \ .$$

So, we have to prove that

$$c \cdot \theta(u, u') \cdot r \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta([\!]_x\, m\, [\!]^y))(x) \ .$$

This is trivial by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$.

For the inductive case where $[\!]_x\, m\,[\!]^y$ is a metablock $[\!]_x\, p \cdot q\,[\!]^y$, we suppose that $u$ is the fresh label used by $\mathsf{mb}$. So, we suppose that

$$\mathsf{mb}(\overset{y}{\underset{x}{[\!]}}\, p \cdot q\, \overset{y}{[\!]}) \quad = \quad \mathsf{mb}(\overset{u}{\underset{x}{[\!]}}\, p\, \overset{u}{[\!]}),\ \mathsf{mb}(\overset{y}{\underset{u}{[\!]}}\, q\, \overset{y}{[\!]}) \ .$$

Then, the inductive hypotheses for this case are that every unary blocks $[\!]_{u'}\, m''\,[\!]^u \in \mathsf{mb}([\!]_x\, p\,[\!]^u)^1$ and $[\!]_{u''}\, m'''\,[\!]^y \in \mathsf{mb}([\!]_u\, q\,[\!]^y)^1$, and every binary blocks $[\!]_{t'}\, c' \downarrow_v \uparrow^{v'} r'\,[\!]^u \in \mathsf{mb}([\!]_x\, p\,[\!]^u)^2$ and $[\!]_{t'}\, c'' \downarrow_{v''} \uparrow^{v'''} r''\,[\!]^y \in \mathsf{mb}([\!]_u\, q\,[\!]^y)^2$ are such that

$$m'' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{u}{\underset{x}{[\!]}}\, p\, \overset{u}{[\!]}))(u') \ , \tag{F.6}$$

$$m''' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{u}{[\!]}}\, q\, \overset{y}{[\!]}))(u'') \ , \tag{F.7}$$

$$c' \cdot \theta(v, v') \cdot r' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{u}{\underset{x}{[\!]}}\, p\, \overset{u}{[\!]}))(t) \ , \tag{F.8}$$

$$c'' \cdot \theta(v'', v''') \cdot r'' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{u}{[\!]}}\, q\, \overset{y}{[\!]}))(t') \ . \tag{F.9}$$

Note that, by definition of $\mathsf{mb}$ and the fact that $u \neq y$, no block of $\mathsf{mb}([\!]_x\, p\,[\!]^u)$ can have $y$ as its ending label. So, it is mandatory that the wanted unary block $[\!]_z\, m'\,[\!]^y$ and binary block $[\!]_{z'}\, c \downarrow_w \uparrow^{w'} r\,[\!]^y$ are respectively in $\mathsf{mb}([\!]_u\, q\,[\!]^y)^1$ and $\mathsf{mb}([\!]_u\, q\,[\!]^y)^2$. By inductive hypotheses (F.7) and (F.9), we have that

$$m' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{u}{[\!]}}\, q\, \overset{y}{[\!]}))(z) \ , \tag{F.10}$$

$$c \cdot \theta(w, w') \cdot r \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{u}{[\!]}}\, q\, \overset{y}{[\!]}))(z') \ . \tag{F.11}$$

To prove (5.13) and (5.14), we just have to prove that

$$(\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\!]}}\, p \cdot q\, \overset{y}{[\!]}))(z) = (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{u}{[\!]}}\, q\, \overset{y}{[\!]}))(z) \ ,$$

$$(\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\!]}}\, p \cdot q\, \overset{y}{[\!]}))(z') = (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{u}{[\!]}}\, q\, \overset{y}{[\!]}))(z') \ .$$

This is direct by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$ which states that

$$\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{u}{[\!]}}\, q\, \overset{y}{[\!]}) \subseteq \mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\!]}}\, p \cdot q\, \overset{y}{[\!]}) \ .$$

For the inductive case where $[\!]_x\, m\,[\!]^y$ is a metablock $[\!]_x\, p + q\,[\!]^y$, by the definition of $\mathsf{mb}$ we have that

$$\mathsf{mb}(\overset{y}{\underset{x}{[\!]}}\, p + q\, \overset{y}{[\!]}) \quad = \quad \mathsf{mb}(\overset{y}{\underset{x}{[\!]}}\, p\, \overset{y}{[\!]}),\ \mathsf{mb}(\overset{y}{\underset{x}{[\!]}}\, q\, \overset{y}{[\!]}) \ .$$

Then, the inductive hypotheses for this case are that every unary blocks $[_u\, m''\,]^y \in \mathsf{mb}([_x\, p\,]^y)^1$ and $[_{u'}\, m'''\,]^y \in \mathsf{mb}([_x\, q\,]^y)^1$, and every binary blocks $[_t\, c' \downarrow_v \uparrow^{v'} r'\,]^y \in \mathsf{mb}([_x\, p\,]^y)^2$ and $[_{t'}\, c'' \downarrow_{v''} \uparrow^{v'''} r''\,]^y \in \mathsf{mb}([_x\, q\,]^y)^2$ are such that

$$m'' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,p\,]}}))(u) \ , \tag{F.12}$$

$$m''' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,q\,]}}))(u') \ , \tag{F.13}$$

$$c' \cdot \theta(v, v') \cdot r' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,p\,]}}))(t) \ , \tag{F.14}$$

$$c'' \cdot \theta(v'', v''') \cdot r'' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,q\,]}}))(t') \ . \tag{F.15}$$

We just prove (5.13) since the proof of (5.14) is similar. Let us do a proof by case analysis on $z$. If $z = x$, then the wanted unary block $[_z\, m'\,]^y$ is either in $\mathsf{mb}([_x\, p\,]^y)^1$ or in $\mathsf{mb}([_x\, q\,]^y)^1$ (or both sets). So, by inductive hypotheses (F.12) and (F.13), we have that either

$$m' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,p\,]}}))(z)$$

or

$$m' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,q\,]}}))(z)$$

is true (or both are true). By Kleene algebra, we have that

$$m' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,p\,]}}))(z) + (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,q\,]}}))(z) \ .$$

So, inequation (5.13) follows from the hypothesis that $z = x$ and by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$ which states that

$$\begin{aligned}&(\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,p + q\,]}}))(x)\\ &= (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,p\,]}}))(x) + (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,q\,]}}))(x) \ .\end{aligned}$$

On the other hand, if $z \neq x$, then the wanted unary block $[_z\, m'\,]^y$ is either in $\mathsf{mb}([_x\, p\,]^y)^1$ or in $\mathsf{mb}([_x\, q\,]^y)^1$, but not in both sets. So, by inductive hypotheses (F.12) and (F.13), we have that either

$$m' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,p\,]}}))(z)$$

or

$$m' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,q\,]}}))(z)$$

is true. For the case that $[_z\, m'\,]^y \in \mathsf{mb}([_x\, p\,]^y)^1$, inequation (5.13) follows from the hypothesis that $z \neq x$ and by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$ which states that

$$(\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[\, p + q\,]^y}))(z) = (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[\, p\,]^y}))(z) \ .$$

For the case that $[_z\, m'\,]^y \in \mathsf{mb}([_x\, q\,]^y)^1$, inequation (5.13) follows from the hypothesis that $z \neq x$ and by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$ which states that

$$(\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[\, p + q\,]^y}))(z) = (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[\, q\,]^y}))(z) \ .$$

For the inductive case where $[_x\, m\,]^y$ is a metablock $[_x\, p^*\,]^y$, we suppose that $u$ is the fresh label used by $\mathsf{mb}$. So, we suppose that

$$\mathsf{mb}(\underset{x}{[\, p^*\,]^y}) \quad = \quad \underset{x}{[\, 1\,]^y},\ \mathsf{mb}(\underset{x}{[\, p\,]^y}),\ \mathsf{mb}(\underset{x}{[\, p\,]^u}),\ \mathsf{mb}(\underset{u}{[\, p\,]^u}),\ \mathsf{mb}(\underset{u}{[\, p\,]^y}) \ .$$

Then, some of the inductive hypotheses for this case[1] are that every unary blocks $[_{u'}\, m''\,]^y \in \mathsf{mb}([_x\, p\,]^y)^1$ and $[_{u''}\, m'''\,]^y \in \mathsf{mb}([_u\, p\,]^y)^1$, and every binary blocks $[_t\, c' \downarrow_v \uparrow^{v'} r'\,]^y \in \mathsf{mb}([_x\, p\,]^y)^2$ and $[_{t'}\, c'' \downarrow_{v''} \uparrow^{v'''} r''\,]^y \in \mathsf{mb}([_u\, p\,]^y)^2$ are such that

$$m'' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[\, p\,]^y}))(u') \ , \tag{F.16}$$

$$m''' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{u}{[\, p\,]^y}))(u'') \ , \tag{F.17}$$

$$c' \cdot \theta(v, v') \cdot r' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[\, p\,]^y}))(t) \ , \tag{F.18}$$

$$c'' \cdot \theta(v'', v''') \cdot r'' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{u}{[\, p\,]^y}))(t') \ . \tag{F.19}$$

Note that, by definition of $\mathsf{mb}$ and the fact that $u \neq y$, no block of $\mathsf{mb}([_x\, p\,]^u)$ and $\mathsf{mb}([_u\, p\,]^u)$ can have $y$ as its ending label. So, it is mandatory that the wanted unary block $[_z\, m'\,]^y$ (respectively, binary block $[_{z'}\, c \downarrow_w \uparrow^{w'} r\,]^y$) is in $\{[_x\, 1\,]^y\}$ or in $\mathsf{mb}([_x\, p\,]^y)^1$ or in $\mathsf{mb}([_u\, p\,]^y)^1$ (respectively, in $\mathsf{mb}([_x\, p\,]^y)^2$ or in $\mathsf{mb}([_u\, p\,]^y)^2$).

We just prove (5.13) since the proof of (5.14) is similar. Let us do a proof by case analysis on $z$ and $m'$. If $z = x$ and $m' = 1$, then the wanted unary block $[_z\, m'\,]^y$ is in $\{[_x\, 1\,]^y\}$ and maybe in $\mathsf{mb}([_x\, p\,]^y)^1$. For this case, we must prove that

$$1 \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[\, p^*\,]^y}))(x) \ .$$

[1] Some inductive hypotheses are not necessary for the proof, so they are omitted.

By the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$1 \leqslant ((\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \,\overset{y}{]}))(x))^* \ .$$

This is trivial by Kleene algebra.

If $z = x$ and $m' \neq 1$, then the wanted unary block $[_z\, m' \,]^y$ is in $\mathsf{mb}([_x\, p \,]^y)^1$. Then, by inductive hypothesis (F.16), we have that

$$m' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \,\overset{y}{]}))(x)$$

is true. By Kleene algebra, we have that

$$m' \leqslant ((\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \,\overset{y}{]}))(x))^* \ .$$

So, inequation (5.13) follows from the hypothesis that $z = x$ and by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$ which states that

$$(\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p^* \,\overset{y}{]}))(x) = ((\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \,\overset{y}{]}))(x))^* \ .$$

If $z = u$, then the wanted unary block $[_z\, m' \,]^y$ is in $\mathsf{mb}([_u\, p \,]^y)^1$. Then, by inductive hypothesis (F.17), we have that

$$m' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{u}{[}\, p \,\overset{y}{]}))(u)$$

is true. By Kleene algebra, we have that

$$m' \leqslant ((\mathsf{mb\_vpre\_suffixes}_\theta(\underset{u}{[}\, p \,\overset{u}{]}))(u))^* \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{u}{[}\, p \,\overset{y}{]}))(u) \ .$$

So, inequation (5.13) follows from the hypothesis that $z = u$ and by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$ which states that

$$\begin{aligned}&(\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p^* \,\overset{y}{]}))(u)\\ &= ((\mathsf{mb\_vpre\_suffixes}_\theta(\underset{u}{[}\, p \,\overset{u}{]}))(u))^* \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{u}{[}\, p \,\overset{y}{]}))(u) \ .\end{aligned}$$

If $z \neq x$ and $z \neq u$, then the wanted unary block $[_z\, m' \,]^y$ is either in $\mathsf{mb}([_x\, p \,]^y)^1$ or in $\mathsf{mb}([_u\, p \,]^y)^1$, but cannot be in both sets. So, by inductive hypotheses (F.16) and (F.17), we have that either

$$m' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \,\overset{y}{]}))(z)$$

or

$$m' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{u}{[\![}\, p \,\overset{y}{]\!]}))(z)$$

is true. For the case that $[\![_z\, m' \,]\!]^y \in \mathsf{mb}([\![_x\, p \,]\!]^y)^1$, inequation (5.13) follows from the hypothesis that $z \neq x$ and by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$ which states that

$$(\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[\![}\, p^* \,\overset{y}{]\!]}))(z) = (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[\![}\, p \,\overset{y}{]\!]}))(z) \ .$$

For the case that $[\![_z\, m' \,]\!]^y \in \mathsf{mb}([\![_u\, p \,]\!]^y)^1$, inequation (5.13) follows from the hypothesis that $z \neq u$ and by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$ which states that

$$(\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[\![}\, p^* \,\overset{y}{]\!]}))(z) = (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{u}{[\![}\, p \,\overset{y}{]\!]}))(z) \ .$$

## F.3 Proof of Lemma 5.7

The proof is done by structural induction on $[\![_x\, m \,]\!]^y$. For the base case where $[\![_x\, m \,]\!]^y$ is such that $m \in \Sigma_{\mathbf{i}} \cup \mathsf{Tests}_{\mathbf{B}}$, there is nothing to prove since there cannot exist a unary block or a binary block such that their ending label is not $y$.

For the base case where $[\![_x\, m \,]\!]^y$ is a metablock $[\![_x (c{\downarrow_u}{\uparrow^{u'}}\, r) \,]\!]^y$, there is nothing to prove since there cannot exist a unary block or a binary block such that their ending label is not $y$.

For the inductive case where $[\![_x\, m \,]\!]^y$ is a metablock $[\![_x\, p \cdot q \,]\!]^y$, we suppose that $v$ is the fresh label used by $\mathsf{mb}$. So, we suppose that

$$\mathsf{mb}(\underset{x}{[\![}\, p \cdot q \,\overset{y}{]\!]}) \quad = \quad \mathsf{mb}(\underset{x}{[\![}\, p \,\overset{v}{]\!]}),\ \mathsf{mb}(\underset{v}{[\![}\, q \,\overset{y}{]\!]}) \ .$$

Then, the inductive hypotheses for this case are that every unary blocks $[\![_{t_1}\, m'' \,]\!]^{t'_1} \in \mathsf{mb}([\![_x\, p \,]\!]^v)^1$ and $[\![_{t_2}\, m''' \,]\!]^{t'_2} \in \mathsf{mb}([\![_v\, q \,]\!]^y)^1$ such that $t'_1 \neq v$ and $t'_2 \neq y$, and binary blocks $[\![_{t''_1}\, c' {\downarrow_{w_1}}{\uparrow^{w'_1}}\, r' \,]\!]^{t'''_1} \in \mathsf{mb}([\![_x\, p \,]\!]^v)^2$ and $[\![_{t''_2}\, c'' {\downarrow_{w_2}}{\uparrow^{w'_2}}\, r'' \,]\!]^{t'''_2} \in \mathsf{mb}([\![_v\, q \,]\!]^y)^2$ such that $t'''_1 \neq v$ and $t'''_2 \neq y$, are such that

$$m'' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[\![}\, p \,\overset{v}{]\!]}))(t'_1) \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[\![}\, p \,\overset{v}{]\!]}))(t_1) \ , \tag{F.20}$$

$$m''' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{[\![}\, q \,\overset{y}{]\!]}))(t'_2) \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{[\![}\, q \,\overset{y}{]\!]}))(t_2) \ , \tag{F.21}$$

$$\begin{aligned} c' \cdot \theta(w_1, w'_1) \cdot r' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[\![}\, p \,\overset{v}{]\!]}))(t'''_1) \\ \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[\![}\, p \,\overset{v}{]\!]}))(t''_1) \ , \end{aligned} \tag{F.22}$$

$$c'' \cdot \theta(w_2, w_2') \cdot r'' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} q \overset{y}{\underset{v}{]}}))(t_2''') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} q \overset{y}{\underset{v}{]}}))(t_2'') \ . \tag{F.23}$$

We just prove (5.15) since the proof of (5.16) is similar. Note that, by definition of $\mathsf{mb}$ and the fact that $v \neq x$ and $v \neq y$, the wanted unary block $[_z\, m'\,]^{z'}$ is either in $\mathsf{mb}([_x\, p\,]^v)^1$ or in $\mathsf{mb}([_v\, q\,]^y)^1$, but cannot be in both sets. Let us do a proof by case analysis. If $[_z\, m'\,]^{z'} \in \mathsf{mb}([_x\, p\,]^v)^1$ and $z' = v$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}} p \cdot q \overset{y}{\underset{x}{]}}))(v) \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}} p \cdot q \overset{y}{\underset{x}{]}}))(z) \ .$$

By the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} q \overset{y}{\underset{v}{]}}))(v) \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{x}{[}} p \overset{v}{\underset{x}{]}}))(z) \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} q \overset{y}{\underset{v}{]}}))(v) \ .$$

By the monotonicity of $\cdot$, it suffices to prove that

$$m' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{x}{[}} p \overset{v}{\underset{x}{]}}))(z) \ .$$

This is direct from Lemma 5.6, inequation (5.13).

If $[_z\, m'\,]^{z'} \in \mathsf{mb}([_x\, p\,]^v)^1$ and $z' \neq v$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}} p \cdot q \overset{y}{\underset{x}{]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}} p \cdot q \overset{y}{\underset{x}{]}}))(z) \ .$$

By the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{x}{[}} p \overset{v}{\underset{x}{]}}))(z') \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} q \overset{y}{\underset{v}{]}}))(v) \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{x}{[}} p \overset{v}{\underset{x}{]}}))(z) \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} q \overset{y}{\underset{v}{]}}))(v) \ .$$

By the monotonicity of $\cdot$, it suffices to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{x}{[}} p \overset{v}{\underset{x}{]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{x}{[}} p \overset{v}{\underset{x}{]}}))(z) \ .$$

This is direct from inductive hypothesis (F.20).

If $[_z\, m'\,]^{z'} \in \mathsf{mb}([_v\, q\,]^y)^1$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}} p \cdot q \overset{y}{\underset{x}{]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}} p \cdot q \overset{y}{\underset{x}{]}}))(z) \ .$$

By the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{\overset{y}{[\,q\,]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{\overset{y}{[\,q\,]}}))(z) \ .$$

This is direct from inductive hypothesis (F.21) since $z' \neq y$ by hypothesis.

For the inductive case where $[_x\, m\,]^y$ is a metablock $[_x\, p + q\,]^y$, we have that

$$\mathsf{mb}(\underset{x}{\overset{y}{[\,p+q\,]}}) \quad = \quad \mathsf{mb}(\underset{x}{\overset{y}{[\,p\,]}}),\ \mathsf{mb}(\underset{x}{\overset{y}{[\,q\,]}}) \ .$$

Then, the inductive hypotheses for this case are that every unary blocks $[_{t_1}\, m''\,]^{t'_1} \in \mathsf{mb}([_x\, p\,]^y)^1$ and $[_{t_2}\, m'''\,]^{t'_2} \in \mathsf{mb}([_x\, q\,]^y)^1$ such that $t'_1 \neq y$ and $t'_2 \neq y$, and every binary blocks $[_{t''_1}\, c' \downarrow_{w_1} \uparrow^{w'_1} r'\,]^{t'''_1} \in \mathsf{mb}([_x\, p\,]^y)^2$ and $[_{t''_2}\, c'' \downarrow_{w_2} \uparrow^{w'_2} r''\,]^{t'''_2} \in \mathsf{mb}([_x\, q\,]^y)^2$ such that $t'''_1 \neq y$ and $t'''_2 \neq y$, are such that

$$m'' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\overset{y}{[\,p\,]}}))(t'_1) \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\overset{y}{[\,p\,]}}))(t_1) \ , \tag{F.24}$$

$$m''' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\overset{y}{[\,q\,]}}))(t'_2) \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\overset{y}{[\,q\,]}}))(t_2) \ , \tag{F.25}$$

$$\begin{aligned} c' \cdot \theta(w_1, w'_1) \cdot r' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\overset{y}{[\,p\,]}}))(t'''_1) \\ \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\overset{y}{[\,p\,]}}))(t''_1) \ , \end{aligned} \tag{F.26}$$

$$\begin{aligned} c'' \cdot \theta(w_2, w'_2) \cdot r'' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\overset{y}{[\,q\,]}}))(t'''_2) \\ \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\overset{y}{[\,q\,]}}))(t''_2) \ . \end{aligned} \tag{F.27}$$

We just prove (5.15) since the proof of (5.16) is similar. Note that, by definition of $\mathsf{mb}$ and the fact that $z' \neq y$, the wanted unary block $[_z\, m'\,]^{z'}$ is either in $\mathsf{mb}([_x\, p\,]^y)^1$ or in $\mathsf{mb}([_x\, q\,]^y)^1$, but cannot be in both sets. Note also that $z' \neq x$. Let us do a proof by case analysis. If $[_z\, m'\,]^{z'} \in \mathsf{mb}([_x\, p\,]^y)^1$ and $z = x$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\overset{y}{[\,p+q\,]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\overset{y}{[\,p+q\,]}}))(x) \ .$$

By the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$\begin{aligned} & m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\overset{y}{[\,p\,]}}))(z') \\ & \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\overset{y}{[\,p\,]}}))(x) + (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\overset{y}{[\,q\,]}}))(x) \ . \end{aligned}$$

By Kleene algebra, it suffices to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,p\,]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,p\,]}}))(x) \ .$$

This is direct from inductive hypothesis (F.24).

If $[_z\, m'\, ]^{z'} \in \mathsf{mb}([_x\, p\, ]^y)^1$ and $z \neq x$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,p+q\,]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,p+q\,]}}))(z) \ .$$

By the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,p\,]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,p\,]}}))(z) \ .$$

This is direct from inductive hypothesis (F.24).

If $[_z\, m'\, ]^{z'} \in \mathsf{mb}([_x\, q\, ]^y)^1$ and $z = x$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,p+q\,]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,p+q\,]}}))(x) \ .$$

By the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$\begin{aligned} &m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,q\,]}}))(z') \\ &\leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,p\,]}}))(x) + (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,q\,]}}))(x) \ . \end{aligned}$$

By Kleene algebra, it suffices to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,q\,]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,q\,]}}))(x) \ .$$

This is direct from inductive hypothesis (F.25).

If $[_z\, m'\, ]^{z'} \in \mathsf{mb}([_x\, q\, ]^y)^1$ and $z \neq x$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,p+q\,]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,p+q\,]}}))(z) \ .$$

By the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,q\,]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\,q\,]}}))(z) \ .$$

This is direct from inductive hypothesis (F.25).

For the inductive case where $\lfloor_x\, m\,\rfloor^y$ is a metablock $\lfloor_x\, p^*\,\rfloor^y$, we suppose that $v$ is the fresh label used by $\mathsf{mb}$. So, we suppose that

$$\mathsf{mb}(\underset{x}{\lfloor}\, p^*\,\overset{y}{\rfloor}) \quad = \quad \underset{x}{\lfloor}\, 1\,\overset{y}{\rfloor},\ \mathsf{mb}(\underset{x}{\lfloor}\, p\,\overset{y}{\rfloor}),\ \mathsf{mb}(\underset{x}{\lfloor}\, p\,\overset{v}{\rfloor}),\ \mathsf{mb}(\underset{v}{\lfloor}\, p\,\overset{v}{\rfloor}),\ \mathsf{mb}(\underset{v}{\lfloor}\, p\,\overset{y}{\rfloor})\ .$$

Then, the inductive hypotheses for this case are that every unary blocks $\lfloor_{t_1}\, m''\,\rfloor^{t'_1} \in \mathsf{mb}(\lfloor_x\, p\,\rfloor^y)^1$, $\lfloor_{t_2}\, m'''\,\rfloor^{t'_2} \in \mathsf{mb}(\lfloor_x\, p\,\rfloor^v)^1$, $\lfloor_{t_3}\, m''''\,\rfloor^{t'_3} \in \mathsf{mb}(\lfloor_v\, p\,\rfloor^v)^1$ and $\lfloor_{t_4}\, m'''''\,\rfloor^{t'_4} \in \mathsf{mb}(\lfloor_v\, p\,\rfloor^y)^1$ such that $t'_1 \neq y$, $t'_2 \neq v$, $t'_3 \neq v$ and $t'_4 \neq y$, and every binary blocks $\lfloor_{t''_1}\, c' \downarrow_{w_1} \uparrow^{w'_1} r'\,\rfloor^{t'''_1} \in \mathsf{mb}(\lfloor_x\, p\,\rfloor^y)^2$, $\lfloor_{t''_2}\, c'' \downarrow_{w_2} \uparrow^{w'_2} r''\,\rfloor^{t'''_2} \in \mathsf{mb}(\lfloor_x\, p\,\rfloor^v)^2$, $\lfloor_{t''_3}\, c''' \downarrow_{w_3} \uparrow^{w'_3} r'''\,\rfloor^{t'''_3} \in \mathsf{mb}(\lfloor_v\, p\,\rfloor^v)^2$ and $\lfloor_{t''_4}\, c'''' \downarrow_{w_4} \uparrow^{w'_4} r''''\,\rfloor^{t'''_4} \in \mathsf{mb}(\lfloor_v\, p\,\rfloor^y)^2$ such that $t'''_1 \neq y$, $t'''_2 \neq v$, $t'''_3 \neq v$ and $t'''_4 \neq y$, are such that

$$m'' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\lfloor}\, p\,\overset{y}{\rfloor}))(t'_1) \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\lfloor}\, p\,\overset{y}{\rfloor}))(t_1)\ , \tag{F.28}$$

$$m''' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\lfloor}\, p\,\overset{v}{\rfloor}))(t'_2) \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\lfloor}\, p\,\overset{v}{\rfloor}))(t_2)\ , \tag{F.29}$$

$$m'''' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{\lfloor}\, p\,\overset{v}{\rfloor}))(t'_3) \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{\lfloor}\, p\,\overset{v}{\rfloor}))(t_3)\ , \tag{F.30}$$

$$m''''' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{\lfloor}\, p\,\overset{y}{\rfloor}))(t'_4) \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{\lfloor}\, p\,\overset{y}{\rfloor}))(t_4)\ , \tag{F.31}$$

$$\begin{aligned} c' \cdot \theta(w_1, w'_1) \cdot r' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\lfloor}\, p\,\overset{y}{\rfloor}))(t'''_1) \\ \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\lfloor}\, p\,\overset{y}{\rfloor}))(t''_1)\ , \end{aligned} \tag{F.32}$$

$$\begin{aligned} c'' \cdot \theta(w_2, w'_2) \cdot r'' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\lfloor}\, p\,\overset{v}{\rfloor}))(t'''_2) \\ \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\lfloor}\, p\,\overset{v}{\rfloor}))(t''_2)\ , \end{aligned} \tag{F.33}$$

$$\begin{aligned} c''' \cdot \theta(w_3, w'_3) \cdot r''' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{\lfloor}\, p\,\overset{v}{\rfloor}))(t'''_3) \\ \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{\lfloor}\, p\,\overset{v}{\rfloor}))(t''_3)\ , \end{aligned} \tag{F.34}$$

$$\begin{aligned} c'''' \cdot \theta(w_4, w'_4) \cdot r'''' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{\lfloor}\, p\,\overset{y}{\rfloor}))(t'''_4) \\ \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{\lfloor}\, p\,\overset{y}{\rfloor}))(t''_4)\ . \end{aligned} \tag{F.35}$$

We just prove (5.15) since the proof of (5.16) is similar. Note that the wanted unary block $\lfloor_z\, m'\,\rfloor^{z'}$ cannot be in $\{\lfloor_x\, 1\,\rfloor^y\}$, because $z' \neq y$ by hypothesis. Note also that, by definition of $\mathsf{mb}$ and the fact that $v \neq x$ and $v \neq y$, the wanted unary block $\lfloor_z\, m'\,\rfloor^{z'}$ is either in $\mathsf{mb}(\lfloor_x\, p\,\rfloor^y)^1$, in $\mathsf{mb}(\lfloor_x\, p\,\rfloor^v)^1$, in $\mathsf{mb}(\lfloor_v\, p\,\rfloor^v)^1$ or in $\mathsf{mb}(\lfloor_v\, p\,\rfloor^y)^1$, but cannot be in

two sets at the same time. Let us do a proof by case analysis. If $[_z\, m'\,]^{z'} \in \mathsf{mb}([_x\, p\,]^y)^1$ and $z = x$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\, p^*\,]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\, p^*\,]}}))(x) \ .$$

By the fact that $z'$ cannot be $x$ and by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\, p\,]}}))(z') \leqslant ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\, p\,]}}))(x))^* \ .$$

By Kleene algebra, it suffices to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\, p\,]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\, p\,]}}))(x) \ .$$

This is direct from inductive hypothesis (F.28) since $z' \neq y$ by hypothesis.

If $[_z\, m'\,]^{z'} \in \mathsf{mb}([_x\, p\,]^y)^1$ and $z \neq x$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\, p^*\,]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\, p^*\,]}}))(z) \ .$$

By the fact that $z'$ cannot be $x$, by the hypothesis that $z \neq x$ and by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\, p\,]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\, p\,]}}))(z) \ .$$

This is direct from inductive hypothesis (F.28).

If $[_z\, m'\,]^{z'} \in \mathsf{mb}([_x\, p\,]^v)^1$, $z = x$ and $z' \neq v$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\, p^*\,]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\, p^*\,]}}))(x) \ .$$

By the fact that $z'$ cannot be $x$ or $v$ and by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$\begin{aligned} m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{x}{[\, p\,]}}))(z') \cdot ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[\, p\,]}}))(v))^* \\ \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[\, p\,]}}))(v) \\ \leqslant ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[\, p\,]}}))(x))^* \ . \end{aligned}$$

By the hypothesis $z' \neq v$, by the inductive hypothesis (F.29) and by Kleene algebra, it suffices to prove that

$$\begin{aligned}&(\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \,\overset{v}{]}))(x) \cdot ((\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{[}\, p \,\overset{v}{]}))(v))^* \\ &\qquad \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{[}\, p \,\overset{y}{]}))(v) \\ &\leqslant ((\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \,\overset{y}{]}))(x))^* \ .\end{aligned}$$

By Lemma 5.5, it suffices to prove that

$$\mathsf{mb\_vpre}_\theta(p) \cdot (\mathsf{mb\_vpre}_\theta(p))^* \cdot \mathsf{mb\_vpre}_\theta(p) \leqslant (\mathsf{mb\_vpre}_\theta(p))^* \ .$$

This is trivial by Kleene algebra.

If $[_z\, m' \,]^{z'} \in \mathsf{mb}([_x\, p \,]^v)^1$, $z = x$ and $z' = v$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p^* \,\overset{y}{]}))(v) \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p^* \,\overset{y}{]}))(x) \ .$$

By the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$\begin{aligned}&m' \cdot ((\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{[}\, p \,\overset{v}{]}))(v))^* \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{[}\, p \,\overset{y}{]}))(v) \\ &\leqslant ((\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \,\overset{y}{]}))(x))^* \ .\end{aligned}$$

By Lemma 5.6, inequation (5.13), it suffices to prove that

$$\begin{aligned}&(\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \,\overset{v}{]}))(x) \cdot ((\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{[}\, p \,\overset{v}{]}))(v))^* \\ &\qquad \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{[}\, p \,\overset{y}{]}))(v) \\ &\leqslant ((\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p \,\overset{y}{]}))(x))^* \ .\end{aligned}$$

By Lemma 5.5, it suffices to prove that

$$\mathsf{mb\_vpre}_\theta(p) \cdot (\mathsf{mb\_vpre}_\theta(p))^* \cdot \mathsf{mb\_vpre}_\theta(p) \leqslant (\mathsf{mb\_vpre}_\theta(p))^* \ .$$

This is trivial by Kleene algebra.

If $[_z\, m' \,]^{z'} \in \mathsf{mb}([_x\, p \,]^v)^1$, $z \neq x$ and $z' \neq v$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p^* \,\overset{y}{]}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{[}\, p^* \,\overset{y}{]}))(z) \ .$$

By the fact that $z$ and $z'$ cannot be $x$ or $v$ and by the definition of the function $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$\begin{aligned} & m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{x}{[}} p \overset{v}{]}))(z') \cdot ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{]}))(v))^* \\ & \quad \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} p \overset{v}{]}))(v) \\ \leqslant\ & (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{x}{[}} p \overset{v}{]}))(z) \cdot ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{]}))(v))^* \\ & \quad \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} p \overset{v}{]}))(v) \ . \end{aligned}$$

By the monotonicity of $\cdot$, it suffices to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{x}{[}} p \overset{v}{]}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{x}{[}} p \overset{v}{]}))(z) \ .$$

This is direct from inductive hypothesis (F.29).

If $[_z\, m'\,]^{z'} \in \mathsf{mb}([_x\, p\,]^v)^1$, $z \neq x$ and $z' = v$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}} p^* \overset{y}{]}))(v) \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}} p^* \overset{y}{]}))(z) \ .$$

By the fact that $z$ cannot be $x$ or $v$ and by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$\begin{aligned} & m' \cdot ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{]}))(v))^* \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} p \overset{v}{]}))(v) \\ \leqslant\ & (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{x}{[}} p \overset{v}{]}))(z) \cdot ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{]}))(v))^* \\ & \quad \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} p \overset{v}{]}))(v) \ . \end{aligned}$$

By the monotonicity of $\cdot$, it suffices to prove that

$$m' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{x}{[}} p \overset{v}{]}))(z) \ .$$

This is direct from Lemma 5.6, inequation (5.13).

If $[_z\, m'\,]^{z'} \in \mathsf{mb}([_v\, p\,]^v)^1$, $z = v$ and $z' \neq v$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}} p^* \overset{y}{]}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}} p^* \overset{y}{]}))(v) \ .$$

By the fact that $z'$ cannot be $x$ or $v$ and by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$\begin{aligned} & m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(z') \cdot ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v))^* \\ & \qquad \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v) \\ & \leqslant ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v))^* \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v) \ . \end{aligned}$$

By the hypothesis $z' \neq v$, by the inductive hypothesis (F.30) and by Kleene algebra, it suffices to prove that

$$\begin{aligned} & (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v) \cdot ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v))^* \\ & \qquad \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v) \\ & \leqslant ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v))^* \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v) \ . \end{aligned}$$

This is trivial by Kleene algebra.

If $[_z\, m'\, ]^{z'} \in \mathsf{mb}([_v\, p\, ]^v)^1$, $z = v$ and $z' = v$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}} p^* \overset{y}{\underset{x}{]}}))(v) \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}} p^* \overset{y}{\underset{x}{]}}))(v) \ .$$

By the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$\begin{aligned} & m' \cdot ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v))^* \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v) \\ & \leqslant ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v))^* \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v) \ . \end{aligned}$$

By Lemma 5.6, inequation (5.13), and by Kleene algebra, it suffices to prove that

$$\begin{aligned} & (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v) \cdot ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v))^* \\ & \qquad \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v) \\ & \leqslant ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v))^* \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v) \ . \end{aligned}$$

This is trivial by Kleene algebra.

If $[_z\, m'\, ]^{z'} \in \mathsf{mb}([_v\, p\, ]^v)^1$, $z \neq v$ and $z' \neq v$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}} p^* \overset{y}{\underset{x}{]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}} p^* \overset{y}{\underset{x}{]}}))(z) \ .$$

By the fact that $z$ and $z'$ cannot be $x$ or $v$ and by the definition of the function $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$\begin{aligned} & m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(z') \cdot ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v))^* \\ & \quad \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v) \\ \leqslant\ & (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(z) \cdot ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v))^* \\ & \quad \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v) \ . \end{aligned}$$

By monotonicity of $\cdot$, it suffices to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(z) \ .$$

This is direct from inductive hypothesis (F.30).

If $[_z\, m'\,]^{z'} \in \mathsf{mb}([_v\, p\,]^v)^1$, $z \neq v$ and $z' = v$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}} p^* \overset{y}{\underset{x}{]}}))(v) \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}} p^* \overset{y}{\underset{x}{]}}))(z) \ .$$

By the fact that $z$ cannot be $x$ or $v$ and by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$\begin{aligned} & m' \cdot ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v))^* \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v) \\ \leqslant\ & (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(z) \cdot ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v))^* \\ & \quad \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v) \ . \end{aligned}$$

By monotonicity of $\cdot$, it suffices to prove that

$$m' \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(z) \ .$$

This is direct from Lemma 5.6, inequation (5.13).

If $[_z\, m'\,]^{z'} \in \mathsf{mb}([_v\, p\,]^y)^1$ and $z = v$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}} p^* \overset{y}{\underset{x}{]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{x}{[}} p^* \overset{y}{\underset{x}{]}}))(v) \ .$$

By the fact that $z'$ cannot be $x$ or $v$ and by the definition of $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$\begin{aligned} & m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} p \overset{y}{\underset{v}{]}}))(z') \\ \leqslant\ & ((\mathsf{mb\_vpre\_suffixes}_\theta(\overset{v}{\underset{v}{[}} p \overset{v}{\underset{v}{]}}))(v))^* \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\overset{y}{\underset{v}{[}} p \overset{y}{\underset{v}{]}}))(v) \ . \end{aligned}$$

By Kleene algebra, it suffices to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{\overset{y}{[}}\, p \,\underset{v}{\overset{y}{]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{\overset{y}{[}}\, p \,\underset{v}{\overset{y}{]}}))(v) \ .$$

This is direct from inductive hypothesis (F.31) since $z' \neq y$ by hypothesis.

If $[_z\, m' \,]^{z'} \in \mathsf{mb}([_v\, p \,]^y)^1$ and $z \neq v$, then we have to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\overset{y}{[}}\, p^* \,\underset{x}{\overset{y}{]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{x}{\overset{y}{[}}\, p^* \,\underset{x}{\overset{y}{]}}))(z) \ .$$

By the fact that $z$ and $z'$ cannot be $x$ or $v$ and by the definition of the function $\mathsf{mb\_vpre\_suffixes}_\theta$, it suffices to prove that

$$m' \cdot (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{\overset{y}{[}}\, p \,\underset{v}{\overset{y}{]}}))(z') \leqslant (\mathsf{mb\_vpre\_suffixes}_\theta(\underset{v}{\overset{y}{[}}\, p \,\underset{v}{\overset{y}{]}}))(z) \ .$$

This is direct from inductive hypothesis (F.31) since $z' \neq y$ by hypothesis.