Technical University of Denmark

Model Checking as Static Analysis

Zhang, Fuyuan; Nielson, Flemming; Nielson, Hanne Riis

Publication date:
2012

Document Version
Publisher's PDF, also known as Version of record

Citation (APA):
Zhang, F., Nielson, F., & Nielson, H. R. (2012). Model Checking as Static Analysis. Kgs. Lyngby: Technical University of Denmark (DTU). (IMM-PHD-2012; No. 280).

Model Checking as Static Analysis

Fuyuan Zhang

Kongens Lyngby 2012
IMM-PhD-2012-280

Technical University of Denmark
Informatics and Mathematical Modelling
Building 321, DK-2800 Kongens Lyngby, Denmark
Phone +45 45253351, Fax +45 45882673
reception@imm.dtu.dk
www.imm.dtu.dk IMM-PhD-2012-280

Summary (English)

Both model checking and static analysis are prominent approaches to detecting software errors. Model Checking is a successful formal method for verifying properties specified in temporal logics with respect to transition systems. Static analysis is also a powerful method for validating program properties which can predict safe approximations to program behaviors. In this thesis, we have developed several static analysis based techniques to solve model checking problems, aiming at showing the link between static analysis and model checking.

We focus on logical approaches to static analysis. Alternation-free Least Fixed Point Logic (ALFP), an extension of Datalog, has been used as the specification language in most of our research results.

We have first considered the CTL model checking and developed an ALFP-based technique to solve the CTL model checking problem. We have shown that the set of states satisfying a CTL formula can be characterized as the least model of ALFP clauses specifying this CTL formula. The existence of the least model of ALFP clauses is ensured by the Moore Family property of ALFP. Then, we take fairness assumptions in CTL into consideration and have shown that CTL fairness problems can be encoded into ALFP as well.

To deal with multi-valued model checking problems, we have proposed multi-valued ALFP. A Moore Family result for multi-valued ALFP is also established, which ensures the existence and uniqueness of the least model. When the truth

values in multi-valued ALFP constitute a finite distributive complete lattice, multi-valued ALFP can be reduced to two-valued ALFP. This result enables to implement a solver for multi-valued ALFP by reusing existing solvers for two-valued ALFP. Our ALFP-based technique developed for the two-valued CTL naturally generalizes to a multi-valued setting, and we therefore obtain a multi-valued analysis for temporal properties specified by CTL formulas. In particular, we have shown that the three-valued CTL model checking problem over Kripke modal transition systems can be exactly encoded in three-valued ALFP.

Last, we come back to two-valued settings and have considered the model checking for the modal μ-calculus. Our results have shown that ALFP suffices to deal with the model checking problem for the alternation-free μ-calculus. However, to deal with the full fragment of the μ-calculus, we need to go beyond ALFP. Therefore, we proposed Succinct Fixed Point Logic (SFP), as an extension of ALFP. We have established a Moore Family result for SFP, which ensures the existence and uniqueness of the intended model of SFP. We have shown that SFP is well suited to specify nested fixed points in the μ-calculus and the model checking problem for the μ-calculus can be encoded as the intended model of SFP.

Our research results have strengthened the link between model checking and static analysis. This provides a theoretical foundation for developing a unified tool for both model checking and static analysis techniques.

Summary (Danish)

Både model tjek og statisk analyse kan med succes bruges til at finde fejl i software. Model tjek er en formel metode til at validere egenskaber specificeret i en modal logik mod en model i form af et transitionssystem. Statisk analyse er en udbredt metode til sikkert at approksimere programmers opførsel. I denne afhandling udvikles en række statiske analyser til at foretage model tjek, for herigennem at demonstrere den kraftige forbindelse mellem model tjek og statisk analyse.

Udviklingen baserer sig på logiske tilgangsvinkler til statisk analyse og tager konkret udgangspunkt i “Alternation-free Least Fixed Point Logic (ALFP)", der er en udvidelse af Datalog.

Vi studerer først model tjek af den modale logik “Computation Tree Logic (CTL)"og udvikler en ALFP-baseret løsning af dette. Vi viser at mængden af tilstande, der opfylder en CTL formel, kan karakteriseres som den mindste model for de ALFP formler, der modsvarer CTL formlen. Den såkaldte “Moore Family"egenskab ved ALFP formler sikrer eksistensen af en mindste model. Dernæst betragter vi CTL formler under antagelse af fairness og viser at også denne problemstilling kan kodes i ALFP.

Vi udvikler derefter en version af ALFP med mange logiske værdier kaldet “multi-valued ALFP". Vi beviser at “Moore Family"egenskaben også holder for denne udvidelse og der dermed findes præcis én mindste model. Når de logiske værdier udgør et endeligt gitter med passende egenskaber kan model tjek for “multi-valued ALFP"reduceres til sædvanlig model tjek af ALFP. Dermed kan vi udnytte eksisterende implementationer af ALFP model tjek til også at håndtere

“multi-valued ALFP". Ydermere kan vi generalisere vore ALFP-baserede løsning af CTL model tjek til at give model tjek af en version af CTL med mange logiske værdier og det omfatter tre-værdi CTL model tjek over såkaldte Kripke modale transitionssystemer.

Sidst ser vi på model tjek af den såkaldte “modale μ-kalkule"med de sædvanlige to logiske værdier. Her er der positive resultater for et fragment of den “modale μ-kalkule", hvor de logiske kvantorer ikke må alternere. Men der er negative resultater for den fulde “modale μ-kalkule", og det leder frem til at udvikle “Succinct Fixed Point Logic (SFP)"som en ægte udvidelse af ALFP. Vi beviser et “Moore Family"resultat for denne udvidelse og sikrer derved at der altid er præcis én mindste model. Endeligt viser vi at model tjek af den fulde “modale μ-kalkule"kan kodes i SFP.

Samlet set styrker vore resultater vores viden om samspillet mellem model tjek og statisk analyse. Vi har dermed skabt det teoretiske grundlag for udviklingen af et fælles værktøj for statisk analyse og model tjek.

Preface

This thesis was prepared at the Department of Informatics and Mathematical Modelling at the Technical University of Denmark in fulfilment of the requirements for acquiring a PhD degree in Informatics.

The PhD study has been supervised by Professor Flemming Nielson and Professor Hanne Riis Nielson from September 2009 to August 2012. The PhD project has been funded by MT-LAB, A VKR Center of Excellence in the Modelling of Information Technology, the FIRST PhD School and the DTU Informatics.

Most of the work behind this thesis has been carried out independently and I take full responsibility for its contents. Chapter 4 is based on my work [67] under submission. Chapter 5 and 6 are based on my published work [68, 69], coauthored by my supervisors.

Lyngby, 31-August-2012

Fuyuan Zhang

Acknowledgements

I would like to thank my supervisors Professor Flemming Nielson and Professor Hanne Riis Nielson for providing me with the opportunity to work on such an exciting research topic, for their patience and excellent guidance. I have learned a lot by working with them.

I would like to thank the rest of the LBT group and its former members: Jose Nuno Carvalho Quaresma, Piotr Filipiuk, Alejandro Mario Hernandez, Sebastian Alexander Modersheim, Christian W. Probst, Carroline Dewi Puspa Kencana Ramli, Michal Tomasz Terepeta, Kebin Zeng, Lijun Zhang, Roberto Vigo, Fan Yang, Ender Yuksel, Nataliya Skrypnyuk, Han Gao, Matthieu Queva. I have had a good time in LBT with them.

I would like to thank Professor Edmund M. Clarke for hosting my external research stay in Carnegie Mellon University. I have spent an impressive and fruitful stay there.

I would like to thank the evaluation committee: Michael R.A. Huth, Mads Dam and Christian W. Probst.

Last, I would like to thank my parents for their support and encouragement.

Contents

Chapter 1

Introduction

Model Checking [2, 10] is a successful formal method in verifying properties of systems, and intensive researches have been made since its advent. In the model-checking framework, system properties, specified in temporal logics, are checked automatically by exhaustively exploring all execution paths of the modeled system. Hence, model checking, when an counterexample is witnessed, can detect very intricate and deep violations that are hard for other techniques to find. *Computation Tree Logic (CTL)* [2, 4, 3, 5] and *Linear Temporal Logic (LTL)* [1] are two useful temporal logics in model checking. Significant progress has been made on conquering the state explosion problem. This includes Symbolic Model Checking [83, 85, 84] and Partial Order Reduction [86, 87, 88]. Other basic basic approaches to the state explosion problem include Compositional Reasoning [89, 90, 91, 92], Abstraction [93, 94, 95], Symmetry Reduction [96, 97, 98] and Induction [100, 99, 101].

Static analysis [11] is also a powerful method in validation of program properties. In static analysis technique, information is combined from different parts of the program and safe approximations to program behaviors are predicted. Originally used in the development of compilers, it now has been applied to program validation, program understanding and process calculi as well. Typical static analysis approaches include Data Flow Analysis [70, 71, 102], Control Flow Analysis [72, 73, 103], Abstract Interpretation [62, 63, 104], and Type and Effect Systems [74, 75, 105].

Early works [16, 17, 19, 20] have taken the view that static analysis problems can be reduced to model checking. It is shown in [16, 17] that *data flow analysis* can be specified in a sublanguage of the modal μ-calculus [14] so that data flow equations can be implemented by evaluating a specific model checker. The results in [19, 20] show that data flow analysis can be reduced to model checking of a variant of Computation Tree Logic.

In the other direction, recent research [21] presents a flow logic approach [15] to static analysis which encodes the model checking problem for *Action Computation Tree Logic* [28] formulas in *Alternation-free Least Fixed Point Logic* (ALFP [29]). ALFP is more expressive than Datalog [32, 33] and has been used in a number of papers for specifying static analysis and there are a number of solvers available [51].

Continuing the line of work in [21], we develop static analysis techniques to solve model checking problems. We still focus on logical approaches to specifying the analysis constraints that constitute the static analysis and ALFP has been used as the specification language in most of our work.

We provide some knowledge background of our work in Chapter 2, where we introduce partially ordered set and complete lattices, Alternation-free Least Fixed Point Logic, Computation Tree Logic, and the modal μ-calculus. Chapter 3 to Chapter 6 explain our main research results. Chapter 7 gives our conclusion.

In Chapter 3, we develop ALFP-based techniques to solve model checking problems for CTL. Similar to the work in [21], we develop a flow logic approach to static analysis and encode CTL formulas into ALFP. We first consider the CTL semantics without fairness assumptions. We encode CTL formulas into ALFP formulas and show that the set of states satisfying a CTL formula can be exactly characterized by the least solution to the ALFP formulas encoding this CTL formula. Then, we take one step further and consider the CTL semantics with fairness assumptions. Model checking algorithms in [10] provide a good insight for understanding the fairness problems in CTL, where calculating strongly connected components in transition systems plays an important role. We have considered unconditional, weak and strong fairness constraints introduced in [10] and give corresponding ALFP specifications for each of these problems.

In Chapter 4, we show that it is possible to generalize our ALFP-base techniques to a multi-valued setting. We first develop multi-valued ALFP. In multi-valued ALFP, we introduce more than two truth values and require that these truth values constitute a complete lattice. We establish a Moore family property for multi-valued ALFP as well. We show that multi-valued ALFP can be reduced to two-valued ALFP when the truth values constitute a finite distributive complete lattice. This enables us to implement a solver for multi-valued ALFP by reusing existing solvers for two-valued ALFP.

The two-valued analysis developed for CTL model checking problem naturally generalizes to a multi-valued analysis for CTL over multi-valued transition systems when we interpret those ALFP clauses using multi-valued semantics. Many of the equivalences of CTL formulas in the two-valued setting are preserved in our multi-valued setting. To give an application of our multi-valued analysis, we consider the three-valued CTL model checking problem over Kripke modal transition systems [40, 41, 56]. Our result shows that three-valued ALFP-based analysis can exactly characterize the three-valued CTL model checking problem. Therefore, this also generalizes the work in Chapter 3 and [21].

In Chapter 5 and Chapter 6, we come back to two-valued logics and consider the model checking problem for the modal μ-calculus [2, 14]. This is a more expressive logic than CTL and could encode fairness assumptions of CTL as well [6]. Our research results show that ALFP suffices to encode the alternation-free fragment of the μ-calculus. However, to specify mutually dependent least and greatest fixed points, we propose *Succinct Fixed Point Logic* (SFP) which goes beyond ALFP. A Moore family result for SFP is also established and this shows its link to abstract interpretation.

In Chapter 5, we consider the alternation-free fragment of the μ-calculus. We first propose an Alternation-free Normal Form (AFNF), where negations are only applied to closed subformulas. The expressive power of closed formulas in AFNF is equivalent to the alternation-free fragment of the μ-calculus. It is then shown that model checking for the alternation-free μ-calculus can be encoded in ALFP with the usual notion of *stratification*.

When negations are applied to open μ-calculus subformulas, our ALFP-based encoding method fails. We establish a negative result to show that the least fixed point semantics of some μ-calculus formulas of alternation depth greater and equal to 2 cannot be characterized as a Moore Family property with respect to any notion of ranking. The negative result suggests us to look for a more expressive logic than ALFP.

In Chapter 6, we focus on the full fragment of the μ-calculus. There, we propose *Succinct Fixed Point Logic* (SFP) as an extension of ALFP. To specify nested fixed points, we go beyond the notion of stratification used in ALFP and propose the notion of *weak stratification*. This notion allows us to use a larger fragment of clause sequences to specify analysis constraints. The idea behind it is to characterize the requirement of *syntactic monotonicity* in the syntax of the μ-calculus. To facilitate our development, we explicitly introduce a least fixed point operator in SFP.

The main purpose of SFP is to characterize the fixed point semantics of the μ-calculus. In our setting, this amounts to establish the *intended model* of SFP clause sequences. This is done by defining the semantics of the least fixed point operator that we have introduced. Our result shows that the intended model of an SFP clause sequence specifying a μ-calculus formula exactly characterizes the set of states which satisfy this μ-calculus formula over Kripke structures.

Our work, together with results in [16, 19], has improved our understanding of the link between model checking and static analysis. Our research results provide a theoretical foundation for developing a unified tool for both model checking and static analysis techniques.

Related Topics: The link between model checking and abstract interpretation has been shown in [109]. The relationship between model checking and constraint solving has been studied in [107, 108]. Researches in [34, 35, 36, 37, 38, 39] are good references where the link between model checking and logic programming has been investigated.

Chapter 2

Preliminaries

This chapter covers background knowledge for this thesis. Section 2.1 gives a basic introduction to partially ordered set and complete lattices. Section 2.2 introduces *Alternation-free Least Fixed Point Logic*. Section 2.3 covers *Computation Tree Logic* and Section 2.4 gives basics about the modal μ-calculus.

2.1 Partially Ordered Set

Let L be a set. A *partial ordering* is a binary relation $\sqsubseteq$ on L that is:

1. reflective: $\forall l \in L : l \sqsubseteq l$,
2. transitive: $\forall l_1, l_2, l_3 \in L : l_1 \sqsubseteq l_2$ and $l_2 \sqsubseteq l_3$ imply $l_1 \sqsubseteq l_3$, and
3. anti-symmetric $\forall l_1, l_2 \in L : l_1 \sqsubseteq l_2$ and $l_2 \sqsubseteq l_1$ imply $l_1 = l_2$.

We also write $l_2 \sqsupseteq l_1$ when $l_1 \sqsubseteq l_2$.

Definition 2.1 (Partially Ordered Set) A *partially ordered set* $(L, \sqsubseteq)$ is a set L equipped with a partial ordering $\sqsubseteq$.

Let L be a partially ordered set and $Y \subseteq L$. An element $l \in L$ is an *upper bound* of Y if $\forall l' \in Y : l' \sqsubseteq l$ and is a *lower bound* of Y if $\forall l' \in Y : l \sqsubseteq l'$. A *least upper bound* of Y, denoted as $\bigsqcup Y$, is an upper bound of Y such that $\bigsqcup Y \sqsubseteq l$ whenever l is an upper bound of Y. A *greatest lower bound* of Y, denoted as $\bigsqcap Y$, is a lower bound of Y such that $l \sqsubseteq \bigsqcup Y$ whenever l is a lower bound of Y. Since $\sqsubseteq$ is anti-symmetric, $\bigsqcup Y$ and $\bigsqcap Y$ are unique whenever they exist.

Definition 2.2 (Complete Lattices) A *complete lattice* $L = (L, \sqsubseteq) = (L, \sqsubseteq, \bigsqcup, \bigsqcap, \bot, \top)$ is a partially ordered set $(L, \sqsubseteq)$ such that all subsets have least upper bounds and greatest lower bounds. Moreover, $\bot = \bigsqcup \emptyset = \bigsqcap L$ is the bottom element and $\top = \bigsqcap \emptyset = \bigsqcup L$ is the top element.

Example 2.1 *Let S be a set. Then $L = (\mathcal{P}(S), \subseteq, \bigcup, \bigcap, \emptyset, S)$ is a complete lattice, where $\mathcal{P}(S)$ is the powerset of S.*

Definition 2.3 (Moore family) A *Moore family* is a subset Y of a complete lattice $L = (L, \sqsubseteq)$ that is closed under greatest lower bounds: $\forall Y' \subseteq Y : \bigsqcap Y' \in Y$.

A Moore family is never empty. It always contains a greatest element $\bigsqcap \emptyset$, which equals the top element $\top$ in L, and a least element $\bigsqcap Y$.

A function $f : L_1 \to L_2$ between partially ordered sets $L_1 = (L_1, \sqsubseteq_1)$ and $L_2 = (L_2, \sqsubseteq_2)$ is *monotone* if

$$\forall l, l' \in L_1 : l \sqsubseteq_1 l' \Rightarrow f(l) \sqsubseteq_2 f(l')$$

It is a *distributive* function if

$$\forall l_1, l_2 \in L_1 : f(l_1 \sqcup l_2) = f(l_1) \sqcup f(l_2)$$

Definition 2.4 (Isomorphism) An *isomorphism* from a partially ordered set $(L_1, \sqsubseteq_1)$ to a partially ordered set $L_2 = (L_2, \sqsubseteq_2)$ is a monotone function $\theta : L_1 \to L_2$ such that there exists a monotone function $\theta^{-1} : L_2 \to L_1$ with $\theta \circ \theta^{-1} = id_2$ and $\theta^{-1} \circ \theta = id_1$, where id_i is the identity function over $L_i, i = 1, 2$.

Let $f : L \to L$ be a monotone function on a complete lattice $L = (L, \sqsubseteq, \bigsqcup, \bigsqcap, \bot, \top)$. A fixed point of f is an element $l \in L$ such that $f(l) = l$ and we use

$$Fix(f) = \{l | f(l) = l\}$$

to denote the set of fixed points of f. The function f is *reductive* at l iff $f(l) \sqsubseteq l$ and we use

$$Red(f) = \{l | f(l) \sqsubseteq l\}$$

to denote the set of elements where f is reductive. The function is *extensive* at l iff $f(l) \sqsupseteq l$ and we use

$$Ext(f) = \{l | f(l) \sqsupseteq l\}$$

to denote the set of elements where f is extensive.

In a complete lattice L. A monotone function f always has a *least fixed point* denoted as

$$lfp(f) = \bigsqcap Fix(f)$$

as well as a *greatest fixed point* denoted as

$$gfp(f) = \bigsqcup Fix(f)$$

The following proposition gives a result of a property of fixed points.

Proposition 2.5 (Taski's Fixed Point Theorem) *[81] Let $L = (L, \sqsubseteq) = (L, \sqsubseteq, \bigsqcup, \bigsqcap, \bot, \top)$ be a complete lattice and $f : L \to L$ be a monotone function on L. Then we have:*

$$lfp(f) = \bigsqcap Red(f) \in Fix(f)$$

$$gfp(f) = \bigsqcup Ext(f) \in Fix(f)$$

More introductions on topics covered in this section can be found in [59] and [11].

2.2 Alternation-free Least Fixed Point Logic

Alternation-free Least Fixed Point Logic is more expressive than Datalog [32, 33] and has been used in a number of papers for specifying static analysis. A simple control flow analysis for Discretionary Ambients [66], which is a variant of the Mobile Ambients [65], is provided in [29] to illustrate the use of ALFP for program analysis. The work in [61] shows an example of using ALFP to perform Reaching Definition Analysis [11]. ALFP [29] has proved to be very useful for expressing static analyses in a general form that can easily be implemented.

Given a fixed countable set $\mathcal{X}$ of variables and a finite alphabet $\mathcal{R}$ of predicate symbols, we define the syntax of ALFP as follows.

$$
\begin{array}{lll}
v & ::= & c \mid x \\
pre & ::= & R(v_1, ..., v_n) \mid \neg R(v_1, ..., v_n) \mid pre_1 \wedge pre_2 \\
 & & \mid pre_1 \vee pre_2 \mid \forall x :\ pre \mid \exists x :\ pre \\
cl & ::= & R(v_1, ..., v_n) \mid \mathbf{true} \mid cl_1 \wedge cl_2 \mid pre \Rightarrow cl \mid \forall x :\ cl
\end{array}
$$

The preconditions and clauses are interpreted over a finite and non-empty universe $\mathcal{U}$. The constant c is an element of $\mathcal{U}$, the variable $x \in \mathcal{X}$ ranges over $\mathcal{U}$, and the n-ary relation $R \in \mathcal{R}$ denotes a subset of $\mathcal{U}^n$.

An *occurrence* of a relation R in a clause is a subformula of the form $R(v_1, ..., v_n)$. If it occurs in a precondition and is not negated, it is a *positive use*. If it occurs in a precondition and is negated, i.e. has the form $\neg R(v_1, ..., v_n)$, it is a *negative use*. All other occurrences are *definitions* and often occur to the right of an implication. To ensure the existence of a least model, we shall pay special attention to the negative uses of relations. We restrict ourselves to the *stratified* fragment of clauses. The notion of stratification is given as follows.

A clause cl is *stratified* if there is a number r, an assignment of numbers called ranks $\texttt{rank}_R \in \{0, ..., r\}$ to each relation R, and a way to write the clause cl in the form $\bigwedge_{0 \le i \le r} cl_i$ such that the following holds for all clauses:

- if cl_i contains a definition of R then $\texttt{rank}_R = i$;
- if cl_i contains a positive use of R then $\texttt{rank}_R \le i$; and
- if cl_i contains a negative use of R then $\texttt{rank}_R < i$.

Example 2.2 *The following clause is not in ALFP since it is ruled out by the notion of stratification:*

$$(\forall x : R_1(x) \Rightarrow R_2(x)) \wedge (\forall x : \neg R_2(x) \Rightarrow R_1(x))$$

This is because it is not possible that we have both $\mathit{rank}_{R_1} \leq \mathit{rank}_{R_2}$ *and* $\mathit{rank}_{R_2} < \mathit{rank}_{R_1}$.

The interpretation of ALFP is given in Table 2.1 in terms of satisfaction relations

$$(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \mathit{pre} \qquad \text{and} \qquad (\varrho, \sigma)\ \underline{\mathtt{sat}}\ \mathit{cl}$$

where ϱ is the interpretation of relations and σ is the interpretation of variables. We write $\varrho(R)$ for the set of k-tuples $(a_1, ...a_k)$ from $\mathcal{U}$ associated with the k-ary predicate R, we use $\sigma(x)$ to denote the atom of $\mathcal{U}$ bound to x and $\sigma[x \mapsto a]$ stands for the mapping that is σ except that x is mapped to a. We also treat a constant c as a variable by setting $\sigma(c) = c$.

$(\varrho, \sigma)\ \underline{\mathtt{sat}}\ R(v_1, ..., v_n)$	iff	$(\sigma(v_1), ..., \sigma(v_n)) \in \varrho(R)$
$(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \neg R(v_1, ..., v_n)$	iff	$(\sigma(v_1), ..., \sigma(v_n)) \notin \varrho(R)$
$(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \mathit{pre}_1 \wedge \mathit{pre}_2$	iff	$(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \mathit{pre}_1$ and $(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \mathit{pre}_2$
$(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \mathit{pre}_1 \vee \mathit{pre}_2$	iff	$(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \mathit{pre}_1$ or $(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \mathit{pre}_2$
$(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \forall x:\ \mathit{pre}$	iff	$(\varrho, \sigma[x \mapsto a])\ \underline{\mathtt{sat}}\ \mathit{pre}$ for all $a \in \mathcal{U}$
$(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \exists x:\ \mathit{pre}$	iff	$(\varrho, \sigma[x \mapsto a])\ \underline{\mathtt{sat}}\ \mathit{pre}$ for some $a \in \mathcal{U}$
$(\varrho, \sigma)\ \underline{\mathtt{sat}}\ R(v_1, ..., v_n)$	iff	$(\sigma(v_1), ..., \sigma(v_n)) \in \varrho(R)$
$(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \mathbf{true}$	iff	$\mathbf{true}$
$(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \mathit{cl}_1 \wedge \mathit{cl}_2$	iff	$(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \mathit{cl}_1$ and $(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \mathit{cl}_2$
$(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \mathit{pre} \Rightarrow \mathit{cl}$	iff	$(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \mathit{cl}$ whenever $(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \mathit{pre}$
$(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \forall x:\ \mathit{cl}$	iff	$(\varrho, \sigma[x \mapsto a])\ \underline{\mathtt{sat}}\ \mathit{cl}$ for all $a \in \mathcal{U}$

Table 2.1: Interpretation of ALFP

A clause with no free variables is called *closed*, and in closed clauses the interpretation σ is of no importance. For a fixed interpretation σ_0, when cl is closed, we have that $(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \mathit{cl}$ agrees with $(\varrho, \sigma_0)\ \underline{\mathtt{sat}}\ \mathit{cl}$.

According to the choice of ranks we have made, we define a lexicographic ordering, $\sqsubseteq$, for the interpretations of relations, ϱ, as follows: $\varrho_1 \sqsubseteq \varrho_2$ if there exists a rank $i \in \{0, ..., r\}$ such that

1. $\varrho_1(R) = \varrho_2(R)$ whenever $\mathit{rank}(R) < i$,

2. $\varrho_1(R) \subseteq \varrho_2(R)$ whenever $rank(R) = i$, and

3. either $i = r$ or $\varrho_1(R) \subset \varrho_2(R)$ for some R with $rank(R) = i$.

We define $\varrho_1 \subseteq \varrho_2$ to mean $\varrho_1(R) \subseteq \varrho_2(R)$ for all $R \in \mathcal{R}$.

The set of interpretations of relations constitutes a complete lattice with respect to $\sqsubseteq$. Moreover, we know from [29] that the set of solutions to an ALFP clause constitutes a Moore Family. The Moore Family result of ALFP is given as follows:

Proposition 2.6 *The set $\{\varrho | (\varrho, \sigma_0)\ \underline{\mathtt{sat}}\ cl\}$ is a Moore Family, i.e. is closed under greatest lower bounds, whenever cl is closed and stratified; the greatest lower bound $\sqcap \{\varrho | (\varrho, \sigma_0)\ \underline{\mathtt{sat}}\ cl\}$ is the least model of cl.*

More generally, given ϱ_0 the set $\{\varrho | (\varrho, \sigma_0)\ \underline{\mathtt{sat}}\ cl \wedge \varrho_0 \subseteq \varrho\}$ is a Moore Family and $\sqcap \{\varrho | (\varrho, \sigma_0)\ \underline{\mathtt{sat}}\ cl \wedge \varrho_0 \subseteq \varrho\}$ is the least model.

2.3 Computation Tree Logic

2.3.1 Kripke Structures

A *Kripke structure* over atomic propositions set $\boldsymbol{P}$ is a tuple $M = (S, T, L)$ where S is a finite set of states, $T \subseteq S \times S$ is a total transition relation, and $L : S \to 2^{\boldsymbol{P}}$ labels each state s with the set of true atomic propositions on it.

We also write $s \to s'$ when $T(s, s')$. Since the transition relation is total, for each state s, there is always a successor s' such that $T(s, s')$. A path $\pi = s_0, s_1...$ where $s_i \to s_{i+1} (0 \leq i)$ is always infinite and we use $\pi[k] (0 \leq k)$ to denote the $(k+1)$th state s_k of π. We use $\pi_{fin} = s_0, s_1...s_n$ where $s_i \to s_{i+1} (0 \leq i \leq n-1)$ to denote a finite path fragment and the length $|\pi_{fin}|$ of $\pi_{fin} = s_0, s_1...s_n$ is $n+1$.

Kripke structures can be used to describe the behaviors of finite-state systems. The set of states S captures all possible interesting snapshots of the system. Transition relation characterizes the evolving of system computations and the

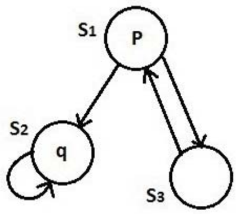


Figure 2.1: Graph Representation of a Kripke structure

function L records some related system properties, described by atomic propositions, when the system is in certain snapshot. A path in a Kripke structure therefore mimics the computations of the system.

Kripke structures can also be represented as graphs. Consider the graph representation of the Kripke structure $M = (S, T, L)$, over atomic propositions set $\boldsymbol{P}$, in Figure 2.1. We know from the figure that $\mathbf{P} = \{p, q\}$ and $S = \{s_1, s_2, s_3\}$. Transition relation T is represented by the edges between states. Function L is represented by the propositions in the circle of each state. For example, neither p nor q is true on state s_3.

2.3.2 Syntax and Semantics of CTL

Computation Tree Logic (CTL) [10, 2] is a branching-time logic. By describing sequences of transitions between states, CTL can be used to specify temporal logic properties about system behaviors without explicitly mentioning time.

We consider the following fragment of CTL where formulas ϕ over a set of propositions $\mathbf{P}$ is defined as follows:

$$\phi ::= \ \mathbf{true} \mid p \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \mathbf{EX}\phi \mid \mathbf{E}[\phi_1 \mathbf{U} \phi_2] \mid \mathbf{AF}\phi$$

where $p \in \mathbf{P}$. This fragment suffices for defining the "remaining operators" using the following equivalences [50]:

$$
\begin{array}{lcl}
\mathbf{AX}\phi & \equiv & \neg\mathbf{EX}\neg\phi \\
\mathbf{EF}\phi & \equiv & \mathbf{E}[\mathbf{true}\mathbf{U}\phi] \\
\mathbf{A}[\phi_1\mathbf{U}\phi_2] & \equiv & \neg\mathbf{E}[\neg\phi_2\mathbf{U}(\neg\phi_1 \wedge \neg\phi_2)] \wedge \mathbf{AF}\phi_2 \\
\mathbf{EG}\phi & \equiv & \neg\mathbf{AF}\neg\phi \\
\mathbf{AG}\phi & \equiv & \neg\mathbf{E}[\mathbf{true}\mathbf{U}\neg\phi]
\end{array}
$$

Symbols **A** and **E** are path quantifiers which mean “for all paths" and “there exists at least one path", respectively. Symbols **X**, **F**, **U** and **G** are temporal operators which mean “next state", “some following state", “Until" and "all following states" respectively.

The semantics of CTL with respect to a Kripke structure is given in Table 2.2. We also write $s \models \phi$ for $(M, s) \models \phi$ when the Kripke structure M is clear from the context.

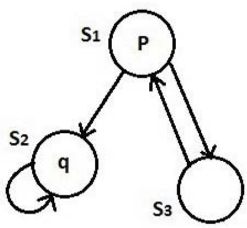
$(M,s) \models \mathbf{true}$	<u>`iff`</u>	*true*
$(M,s) \models p$	<u>`iff`</u>	$p \in L(s)$
$(M,s) \models \neg\phi$	<u>`iff`</u>	$(M,s) \not\models \phi$
$(M,s) \models \phi_1 \wedge \phi_2$	<u>`iff`</u>	$(M,s) \models \phi_1$ `and` $(M,s) \models \phi_2$
$(M,s) \models \mathbf{EX}\phi$	<u>`iff`</u>	`there exists a path` π `from` s `such that` $(M,\pi[1]) \models \phi$
$(M,s) \models \mathbf{E}[\phi_1\mathbf{U}\phi_2]$	<u>`iff`</u>	`there exists a path` π `from` s `such that` $\exists 0 \leq k : (M,\pi[k]) \models \phi_2$ `and` $\forall\ 0 \leq j < k : (M,\pi[j]) \models \phi_1$
$(M,s) \models \mathbf{AF}\phi$	<u>`iff`</u>	`for all paths` π `from` s, $\exists\ 0 \leq k : (M,\pi[k]) \models \phi$

Table 2.2: Semantics for CTL

Model Checking: The problem of *Model Checking* is to find the set of states, on a Kripke structure M, that satisfy a temporal logic formula ϕ ($\{s \mid (M,s) \models \phi\}$) [2]. CTL model checking problem, in which case the temporal logic employed is CTL, can be solved in a syntax directed way. To put it simply, we can calculate the set of states that satisfy each of the subformula of a CTL formula ϕ in a bottom-up manner. We start by handling propositions in the formula ϕ, using the L function in a given Kripke structure. If a state s is labeled with a proposition p, s satisfies p. When handling subformula φ of ϕ, all subformulas of φ should have already been handled. The boolean cases are easy to deal with. For example, assume that we have already computed the set of states S_{ϕ_1} and S_{ϕ_2} which satisfy ϕ_1 and ϕ_2 respectively. To compute the set of states $S_{\phi_1 \vee \phi_2}$

which satisfy $\phi_1 \vee \phi_2$, we have that $S_{\phi_1 \vee \phi_2} = S_{\phi_1} \cup S_{\phi_2}$. The cases for temporal operators are a bit complex and can be found in [2, 10]. Finally we will get the set of states S_ϕ which satisfy our target formula ϕ.

Example 2.3 *Let's consider the problem of finding the set of states which satisfy the CTL formula $\boldsymbol{AF}(p \vee q)$ on the Kripke structure given in the diagram to the left. For each subformula of $\boldsymbol{AF}(p \vee q)$, we calculate the sets of states which satisfy them respectively and list the solutions as follows.*



φ	$\{s \mid s \models \varphi\}$
p	$\{s_1\}$
q	$\{s_2\}$
$p \vee q$	$\{s_1, s_2\}$
$\boldsymbol{AF}(p \vee q)$	$\{s_1, s_2, s_3\}$

The worst case time complexity of CTL model checking is $\mathcal{O}((|T| + |S|)|\phi|)$ [2], where $|\phi|$ is the size of CTL formula, $|T|$ and $|S|$ are the sizes of transition relation and state space of a given Kripke structure respectively.

2.3.3 Fixpoint Representations of CTL

The idea of CTL model checking [79] has been deeply influenced by fixpoint theory such as Tarski's Fixpoint Lemma [81] and Kleene's First Recursion Theorem [82]. Fixpoint representations of CTL can be found in [2, 50]. We first explain the cases of temporal operators **EU** and **AF** briefly in the following. The semantics of each of the two operators can be characterized as the least fixed point of a corresponding monotone function.

For a CTL formula ϕ, we will use $[\![\phi]\!]$ in the following to denote the set of states which satisfy ϕ over Kripke structures.

Case $\mathbf{E}[\phi_1\mathbf{U}\phi_2]$: For a CTL formula $\mathbf{E}[\phi_1\mathbf{U}\phi_2]$, we have the following equivalence according to the semantics of CTL

$$\mathbf{E}[\phi_1\mathbf{U}\phi_2] \equiv \phi_2 \vee (\phi_1 \wedge \mathbf{EXE}[\phi_1\mathbf{U}\phi_2]).$$

From the semantics of the $\mathbf{EX}$ operator, we have that

$$[\![\mathbf{EX}\phi]\!] = \{s | \exists s' : s \to s' \wedge s' \in [\![\phi]\!]\}.$$

From above, we have the following equation

$$[\![\mathbf{E}[\phi_1\mathbf{U}\phi_2]]\!] = [\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \{s | \exists s' : s \to s' \wedge s' \in [\![\mathbf{E}[\phi_1\mathbf{U}\phi_2]]\!]\}).$$

Hence, we can see that $[\![\mathbf{E}[\phi_1\mathbf{U}\phi_2]]\!]$ is a fixed point of the function $F_{\mathbf{E}[\phi_1\mathbf{U}\phi_2]} : \mathcal{P}(S) \to \mathcal{P}(S)$ defined as follows:

$$F_{\mathbf{E}[\phi_1\mathbf{U}\phi_2]}(X) = [\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \{s | \exists s' : s \to s' \wedge s' \in X\}).$$

It is easy to verify that the function $F_{\mathbf{E}[\phi_1\mathbf{U}\phi_2]}$ is monotone. Actually, $[\![\mathbf{E}[\phi_1\mathbf{U}\phi_2]]\!]$ is the least fixed point of this function.

Case $\mathbf{AF}\phi$: For a CTL formula $\mathbf{AF}\phi$, we have the following equivalence according to the semantics of CTL

$$\mathbf{AF}\phi \equiv \phi \vee \mathbf{AXAF}\phi.$$

From the equivalence $\mathbf{AX}\phi \equiv \neg\mathbf{EX}\neg\phi$, we have that

$$[\![\mathbf{AX}\phi]\!] = \{s | \forall s' : s \to s' \texttt{ implies } s' \in [\![\phi]\!]\}.$$

From above, we have the following equation

$$[\![\mathbf{AF}\phi]\!] = [\![\phi]\!] \cup \{s | \forall s' : s \to s' \texttt{ implies } s' \in [\![\mathbf{AF}\phi]\!]\}.$$

Therefore, we know that $[\![\mathbf{AF}\phi]\!]$ is a fixed point of the function $F_{\mathbf{AF}\phi} : \mathcal{P}(S) \to \mathcal{P}(S)$ defined by

$$F_{\mathbf{AF}\phi}(X) = [\![\phi]\!] \cup \{s | \forall s' : s \to s' \texttt{ implies } s' \in X\}.$$

It is easy to see that the function $F_{\mathbf{AF}\phi}$ is a monotone function. In fact, $[\![\mathbf{AF}\phi]\!]$ is its least fixed point.

Then, let us take a look at the **EG** operator. We can derive the semantics of the **EG** operator from the equivalence $\mathbf{EG}\phi \equiv \neg\mathbf{AF}\neg\phi$. Therefore, we have the following:

$$(M, s) \models \mathbf{EG}\phi \quad \underline{\texttt{iff}} \quad \texttt{there exists a path } \pi \texttt{ from } s \texttt{ such that}$$
$$\forall\, 0 \leq k : (M, \pi[k]) \models \phi$$

The semantics of the $\mathbf{EG}\phi$ operator can be characterized as the greatest fixed point of a corresponding monotone function. We explain it as follows.

Case $\mathbf{EG}\phi$: For a CTL formula $\mathbf{EG}\phi$, we have the following equivalence according to the semantics of CTL

$$\mathbf{EG}\phi \equiv \phi \wedge \mathbf{EXEG}\phi.$$

This leads to the following equation

$$[\![\mathbf{EG}\phi]\!] = [\![\phi]\!] \cap \{s | \exists s' : s \rightarrow s' \ \wedge s' \in [\![\mathbf{EG}\phi]\!]\}.$$

Therefore, we know that $[\![\mathbf{EG}\phi]\!]$ is a fixed point of the function $F_{\mathbf{EG}\phi} : \mathcal{P}(S) \rightarrow \mathcal{P}(S)$ defined by

$$F_{\mathbf{EG}\phi}(X) = [\![\phi]\!] \cap \{s | \exists s' : s \rightarrow s' \ \wedge s' \in X\}.$$

It is easy to see that the function $F_{\mathbf{EG}\phi}$ is a monotone function. Actually, $[\![\mathbf{EG}\phi]\!]$ is the greatest fixed point of this function.

2.3.4 CTL with Fairness Assumptions

Fairness assumptions specify fair behaviors over a single computation path and can be used to rule out unrealistic behaviors of the systems modeled by Kripke structures. CTL fairness assumptions are similar to LTL [1] formulas except that CTL state formulas, instead of atomic propositions, are used. A *CTL fairness assumption* is a conjunction of strong, weak and unconditional CTL fairness constraints [10].

We can check whether a CTL fairness assumption is satisfied on a path in Kripke structures. Let π be an infinite path in a given Kripke structure and *fair* be a fixed CTL fairness assumption. The path π is called a *fair path* if π satisfies *fair*. We use notation $\pi \models fair$ to denote this.

An *unconditional CTL fairness constraint* (over $\boldsymbol{P}$) is a term of the form

$$ufair = \bigwedge_{1 \le i \le k} \mathbf{GF}\psi_i$$

where ψ_i is a CTL formula over $\boldsymbol{P}$. As has been introduced before, the symbol $\mathbf{G}$ means "always" and the symbol $\mathbf{F}$ means "in future". Formally, we have the following, where ϕ is a CTL formula and we use notation $\pi \models \mathbf{F}\phi$ (resp. $\pi \models \mathbf{G}\phi$) to mean that ϕ is satisfied on some future states (resp. all of the states) along a path π.

$$\begin{array}{lll} \pi \models \mathbf{F}\phi & \underline{\texttt{iff}} & \exists 0 \le i : (M, \pi[i]) \models \phi \\ \pi \models \mathbf{G}\phi & \underline{\texttt{iff}} & \forall 0 \le i : (M, \pi[i]) \models \phi \end{array}$$

Therefore, $\mathbf{GF}$ means "It is always possible that in future", which can be understood as "infinitely often". Therefore, the constraint $\bigwedge_{1 \le i \le k} \mathbf{GF}\psi_i$ specifies such a path that for each $1 \le i \le k$, the property ψ_i is satisfied on infinitely many states over this path. The formula ψ_i shall be interpreted over states with standard CTL semantics. Formally, we have the following:

$$\pi \models \bigwedge\nolimits_{1 \le i \le k} \mathbf{GF}\psi_i \quad \underline{\texttt{iff}} \quad \forall\, 1 \le i \le k : \forall 0 \le j : \exists j \le j' : (M, \pi[j']) \models \psi_i$$

A *strong CTL fairness constraint* (over $\boldsymbol{P}$) is a term of the form

$$sfair = \bigwedge_{1 \le i \le k} (\mathbf{GF}\phi_i \Rightarrow \mathbf{GF}\psi_i)$$

where ϕ_i and ψ_i are CTL formulas over $\boldsymbol{P}$. The constraint $\bigwedge_{1 \le i \le k}(\mathbf{GF}\phi_i \Rightarrow \mathbf{GF}\psi_i)$ specifies such a path that for each $1 \le i \le k$ if the property ϕ_i is satisfied on infinitely many states over the path, then the property ψ_i shall be satisfied on infinitely many states as well. Formally, we have the following:

$$\begin{array}{lll} \pi \models \bigwedge_{1 \le i \le k}(\mathbf{GF}\phi_i \Rightarrow \mathbf{GF}\psi_i) & \underline{\texttt{iff}} & \forall\, 1 \le i \le k : \texttt{ if } \forall 0 \le j : \exists j \le j' : \\ & & (M, \pi[j']) \models \phi_i \texttt{ then } \forall 0 \le l : \exists l \le l' : \\ & & (M, \pi[l']) \models \psi_i \end{array}$$

Similarly, a *weak CTL fairness constraint* (over $\boldsymbol{P}$) is a term of the form

$$wfair = \bigwedge_{1 \le i \le k} (\mathbf{FG}\phi_i \Rightarrow \mathbf{GF}\psi_i)$$

where **FG** means "finally, it is always the case that". This means the system shall become stable at some point. The constraint $\bigwedge_{1 \le i \le k}(\mathbf{FG}\phi_i \Rightarrow \mathbf{GF}\psi_i)$ means that for each $1 \le i \le k$ if beyond a certain point, the property ϕ_i is satisfied on all the following states, then ψ_i shall be satisfied on infinitely many states along this path. Formally, we have the following:

$$\pi \models \bigwedge_{1 \le i \le k}(\mathbf{FG}\phi_i \Rightarrow \mathbf{GF}\psi_i) \quad \underline{\texttt{iff}} \quad \begin{array}{l} \forall\, 1 \le i \le k:\ \texttt{if}\ \exists 0 \le j : \forall j \le j' : \\ (M, \pi[j']) \models \phi_i\ \texttt{then}\ \forall 0 \le l : \exists l \le l' : \\ (M, \pi[l']) \models \psi_i \end{array}$$

In the following, we introduce *Existential Normal Form* (ENF) for CTL. Each CTL formula can be translated to an equivalent (with respect to $\models$) CTL formula in ENF [10]. We will define the semantics for CTL with fairness assumptions using the syntax of ENF.

Definition 2.7 (Existential Normal Form for CTL) Given $p \in \boldsymbol{P}$, the set of CTL state formulas in existential normal form is defined as follows:

$$\phi ::= \ \mathbf{true} \mid p \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \mathbf{EX}\phi \mid \mathbf{E}[\phi_1 \mathbf{U} \phi_2] \mid \mathbf{EG}\phi$$

In the semantics of CTL with fairness assumptions, path quantifications range over all fair paths rather than over all paths. The semantics of CTL with fairness assumptions are given in Table 2.3. It is actually pointed out in [10] that each CTL formula can be translated to an equivalent CTL formula in ENF with respect to $\models_{fair}$.

The formulas ϕ_i and ψ_i in fairness assumptions are CTL formulas. They are interpreted according to standard CTL semantics without taking into consideration any fairness assumptions. We can use CTL model checking algorithm to determine the set of states which satisfy ϕ_i and ψ_i respectively. Therefore, ϕ_i and ψ_i can be replaced by atomic propositions a_i and b_i. For example, a strong fairness assumption now has this form $fair = \bigwedge_{1 \le i \le k}(\mathbf{GF}a_i \Rightarrow \mathbf{GF}b_i)$.

$(M,s) \models_{fair}$ **true**	<u>`iff`</u>	*true*
$(M,s) \models_{fair} p$	<u>`iff`</u>	$p \in L(s)$
$(M,s) \models_{fair} \neg\phi$	<u>`iff`</u>	$(M,s) \not\models_{fair} \phi$
$(M,s) \models_{fair} \phi_1 \wedge \phi_2$	<u>`iff`</u>	$(M,s) \models_{fair} \phi_1$ `and` $(M,s) \models_{fair} \phi_2$
$(M,s) \models_{fair} \mathbf{EX}\phi$	<u>`iff`</u>	`there exists a fair path` π `from` s `such that` $(M,\pi[1]) \models_{fair} \phi$
$(M,s) \models_{fair} \mathbf{E}[\phi_1 \mathbf{U} \phi_2]$	<u>`iff`</u>	`there exists a fair path` π `from` s `such that` $\exists 0 \leq k : (M,\pi[k]) \models_{fair} \phi_2$ `and` $\forall\ 0 \leq j < k : (M,\pi[j]) \models_{fair} \phi_1$
$(M,s) \models_{fair} \mathbf{EG}\phi$	<u>`iff`</u>	`there exists a fair path` π `from` s `such that` $\forall\ 0 \leq k : (M,\pi[k]) \models_{fair} \phi$

Table 2.3: Semantics for CTL in ENF with Fairness Assumptions

Useful observations are made in [10] to simplify the model checking problem for CTL with fairness constraints. We summarize some of them in the following.

Given a Kripke structure $M = (S,T,L)$. If we remove all the transitions $s \rightarrow s'$ for which either $(M,s) \not\models_{fair} \phi$ or $(M,s') \not\models_{fair} \phi$, we can get a new transition relation T_ϕ. Therefore, we have that $(s,s') \in T_\phi$ if and only if $(s,s') \in T$ and $(M,s) \models_{fair} \phi$ and $(M,s') \models_{fair} \phi$. We use $M_\phi = (S,T_\phi,L)$ to denote the new Kripke structure. Notice that $M_{true} = M$. We have the following fact.

Fact 2.3.1 *Let π be a path in M. It's easy to see that $\forall\ 0 \leq k : (M,\pi[k]) \models_{fair} \phi$ iff π is a path in M_ϕ.*

When considering fair paths, we have the following fact, which means a path is fair iff one of its suffix is fair iff all of its suffixes are fair.

Fact 2.3.2 *For any fairness assumption $fair$, we have that $\pi \models fair$ iff $\pi[j..] \models fair$ for some $j \geq 0$ iff $\pi[j..] \models fair$ for all $j \geq 0$, where $\pi[j..]$ is the suffix of π starting from $\pi[j]$.*

Let *fair* be a fixed CTL fairness assumption. We define $Path^{\phi}_{fair}(s)$ as the set of fair paths in M_ϕ starting from state s. Therefore, $Path^{\phi}_{fair}(s) = \{\pi |\ \pi$ is a path in M_ϕ and $\pi \models fair \wedge \pi[0] = s\}$.

CTL model checking under fairness assumptions can also be handled in a bottom-up manner. When dealing with a formula ϕ, we assume that all the subformulas of ϕ have already been processed and we replaced them with atomic propositions. Boolean fragment of formulas are easy to deal with. Following properties provide useful insight to handle formulas of the forms $\mathbf{EX}p$, $\mathbf{E}[p_1\mathbf{U}p_2]$ and $\mathbf{EG}p$, where p, p_1 and p_2 are atomic propositions. The proofs for the cases of $\mathbf{EX}p$ and $\mathbf{E}[p_1\mathbf{U}p_2]$ can be found in [10]. The case of $\mathbf{EG}p$ is straightforward.

$(M,s) \models_{fair} \mathbf{EX}p$	iff	there exists a state s' such that $s \to s'$, $(M,s') \models p$ and $Path^{true}_{fair}(s') \neq \emptyset$
$(M,s) \models_{fair} \mathbf{E}[p_1\mathbf{U}p_2]$	iff	there exists a finite path fragment $\pi_{fin} = s_0, ..., s_k$ such that $s = s_0$, $(M,s_k) \models p_2, \forall\, 0 \leq j < k : (M,s_j) \models p_1$, and $Path^{true}_{fair}(s_k) \neq \emptyset$
$(M,s) \models_{fair} \mathbf{EG}p$	iff	$Path^{p}_{fair}(s) \neq \emptyset$

Table 2.4: Properties for **EX**, **EU** and **EG** operators under fairness assumptions

It's pointed out in [10] that the model checking problem for CTL with fairness constraints can be reduced to the model checking problem for CTL without fairness and the problem of calculating the set of states $\{s|(M,s) \models_{fair} \mathbf{EG}p\}$ where p is an atomic proposition.

2.4 The Modal μ-calculus

This section introduces basics about the modal μ-calculus [2, 14]. The syntax of the modal μ-calculus is defined as follows.

Definition 2.8 (Syntax of the Modal μ-calculus) Let Var be a set of variables, and $\mathbf{P}$ be a set of atomic propositions. The syntax of the modal μ-calculus formulas is defined as follows:

$$\phi ::= \ p \mid Q \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid \langle a\rangle\phi \mid [a]\phi \mid \mu Q.\phi \mid \nu Q.\phi$$

Here $p \in \mathbf{P}$, $Q \in Var$ and $a \in T$. The μ (resp. ν) operator is the least (resp. greatest) fixed point operator. For $\mu Q.\phi$ and $\nu Q.\phi$, it is required that all occurrences of Q in ϕ are under *an even number of negations* within ϕ. In this case, ϕ is said to be *syntactically monotone* in Q. If a variable is not bound by any fixed point operator in a formula, the variable is called a *free variable.* A formula is *closed* if there are no free variables in it.

A formula ϕ is interpreted as the set of states, on a given Kripke structure, that make it true and this set of states is denoted $[\![\phi]\!]_e$, where $e : Var \to 2^S$ is an environment. Here, the definition of *Kripke structure* is modified slightly in comparison with the one we give in Section 2.3 to distinguish different transitions in a system. Formally, a Kripke structure over a set $\boldsymbol{P}$ of atomic propositions is a tuple $M = (S, T, L)$, where S is a set of states, T is a set of transition relations, and $L : S \to 2^{\boldsymbol{P}}$ labels each state with the set of true atomic propositions. Each element a in T is a transition relation and $a \subseteq S \times S$. We also assume that the Kripke structure is total, although this is not necessary for our development.

We use $e[Q \mapsto W]$ to denote the new environment updated from e by binding the relational variable Q to the set of states $W \subseteq S$. The semantics of the μ-calculus formulas are defined as follows.

- $[\![p]\!]_e = \{s | p \in L(s)\ \}$
- $[\![Q]\!]_e = e(Q)$
- $[\![\neg\phi]\!]_e = S \setminus [\![\phi]\!]_e$
- $[\![\phi_1 \vee \phi_2]\!]_e = [\![\phi_1]\!]_e \cup [\![\phi_2]\!]_e$
- $[\![\phi_1 \wedge \phi_2]\!]_e = [\![\phi_1]\!]_e \cap [\![\phi_2]\!]_e$
- $[\![\langle a\rangle\phi]\!]_e = \{s | \exists s' : (s,\ s') \in a \ \texttt{and}\ s' \in [\![\phi]\!]_e\}$
- $[\![[a]\phi]\!]_e = \{s | \forall s' : (s,\ s') \in a \ \texttt{implies}\ s' \in [\![\phi]\!]_e\}$
- $[\![\mu Q.\phi]\!]_e$ is the least fixpoint of the function $\tau : 2^S \to 2^S$ defined by $\tau(W) = [\![\phi]\!]_{e[Q \mapsto W]}$
- $[\![\nu Q.\phi]\!]_e$ is the greatest fixpoint of the function $\tau : 2^S \to 2^S$ defined by $\tau(W) = [\![\phi]\!]_{e[Q \mapsto W]}$

The boolean operators have the usual meanings. If $(s, s') \in a$, we call s' an a-derivative of s. A state s satisfies $\langle a\rangle\phi$ if some of the a-derivatives of it satisfy

ϕ. A state s satisfies $[a]\phi$ if all a-derivatives of it satisfy ϕ. Notice that if s has no a-derivatives, s satisfies $[a]\phi$ trivially. Due to the restricted use of negations in ϕ, monotonicity is guaranteed [2] for the function $\tau(W) = [\![\phi]\!]_{e[Q \mapsto W]}$.

A formula is in *Positive Normal Form* (PNF) [6] if all negations are only applied to atomic propositions and no variable is quantified twice. We give the syntax of the μ-calculus in Negation-free PNF as follows.

Definition 2.9 (Negation-free PNF for the μ-calculus) Let Var be a set of variables, $\mathbf{P}$ be a set of atomic propositions that is closed under negations. The syntax of the μ-calculus in Negation-free PNF is defined as follows:

$$\phi ::= \; p \mid Q \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid \langle a \rangle \phi \mid [a]\phi \mid \mu Q.\phi \mid \nu Q.\phi$$

where no variable is quantified twice.

It is easy to see that every closed μ-calculus formula can be transformed to its Negation-free PNF provided that the set $\mathbf{P}$ of atomic propositions is closed under negation. To do this, we can push negations as deep as possible using De Morgan's law and the dualities $\neg[a]\phi \equiv \langle a \rangle \neg\phi$, $\neg\langle a \rangle\phi \equiv [a]\neg\phi$, $\neg\mu Q.\phi \equiv \nu Q.\neg\phi[\neg Q/Q]$, and $\neg\nu Q.\phi \equiv \mu Q.\neg\phi[\neg Q/Q]$ and then substitute each negated occurrence $\neg p$ of atomic proposition p with a new atomic proposition p'.

Model checking for the μ-calculus is to find the set of states, on a given Kripke structure, that satisfy the μ-calculus formula ϕ according to the semantics ($[\![\phi]\!]_e$). Efficient algorithms for the μ-calculus model checking problems have been proposed in [6, 110, 8, 9, 7]. Researchers have also proposed local model checking algorithms, which are designed to check whether a specific state of a Kripke structure satisfies a given formula. Efficient local model checking algorithms for the μ-calculus can be found in [8, 12, 13, 18].

CHAPTER 3

CTL in Alternation-free Least Fixed Point Logic

In this chapter, we encode the model checking problem for CTL into ALFP. Along the line of work in [21], we use the flow logic approach to static analysis, where ALFP is used as the specification language. Unlike in [21], we exempt ourself from the heavy labeling skill when developing the flow logic for CTL.

Moreover, we also consider the fairness assumptions for CTL model checking problems. We show that fairness assumptions for CTL can be encoded by ALFP as well. We notice that an exponential blow up occurs when dealing with strong fairness constraints. We mentioned briefly that the exponential blow up could be avoided if we encode it in SFP, introduced in Chapter 6, instead.

The structure of this chapter is as follows. In Section 3.1, we briefly introduce the flow logic and then develop a flow logic approach to encode the CTL model checking problem (without fairness assumptions) into ALFP. Section 3.2 takes CTL fairness assumptions into consideration and shows that we can deal with fairness problems in ALFP as well.

3.1 CTL in ALFP

3.1.1 Flow Logic

Static analysis predicts safe approximations of program behaviors by analyzing information collected from different parts of the program or system. The information is usually represented as elements of complete lattices. Flow logic [15, 31, 22, 23, 24, 25, 26, 27] is an approach to static analysis which is rooted upon existing classical static analysis techniques such as Data Flow Analysis, Constraint Based Analysis, and Abstract Interpretation. It separates the specification and implementation of static analysis and has been applied to the analysis of functional, imperative, and object-oriented programming languages as well as process calculi.

In flow logic approach, for each syntactic category of a program, we define a *logical judgement* expressing the acceptability of the analysis information for that syntactic entity. It requires that our analysis estimate correctly captures the information we have collected from the program. The judgement itself is defined by several constraints expressed in certain languages, i.e first order logic. We must make sure these judgements are well-defined. In general, the judgements are interpreted co-inductively and in syntax-directed definitions it coincides with inductive interpretations. For each syntactic entity, we will use the following clause to specify our static analysis approach.

$$\vec{R} \vdash P \text{ iff } \Psi$$

The judgment $\vec{R} \vdash P$ defined above consists of two elements, the analysis information $\vec{R}$ and the syntactic entity P. $\vec{R}$ usually consists of elements of several complete lattices that are related to the properties we are interested in. In model checking cases, we are interested in the set of states that satisfy a formula on a given Kripke structure. Therefore, when we develop a flow logic approach to solve model checking problems, $\vec{R}$ consists of sets of states. P is one of the syntactic entities of our program. When analyzing CTL, P will be one of those formulas defined in the syntax of CTL. The judgement expresses the validity of our analysis estimates. Constraints Ψ is used to specify the judgment and it constrains the values of the elements of $\vec{R}$. It can be expressed in several languages. In this thesis, we choose to express the constraints in ALFP, which has been introduced before. The clause ($\vec{R} \vdash P$ iff Ψ) means our static analysis estimate $\vec{R}$ is acceptable for the syntactic entity P if and only if the constraint Ψ is satisfied.

When using flow logic approach to analyze a program, we start by defining all the judgements we need. Then, according to the flow logic approach we have developed, we replace all the judgements by the constraints defining them and thus generate all the equivalent constraints which guarantee the correctness of our analysis estimate. This works well in syntax-directed flow logic definitions and all the judgements can be substituted by constraints. Finally, we calculate the least model of our constraints, which is the best analysis results ensured by our Moore Family result. ALFP serves to provide an efficient way of calculating the analysis estimate we need.

3.1.2 Encoding CTL in ALFP

In this section, we consider how to use flow logic to encode CTL formulas into ALFP. Section 2.3.3 has provided semantic insights into CTL temporal operators. Our encoding is directly based on those insights. The syntax of CTL we consider here is the one given in Section 2.3.2. There are mainly two reasons why we choose this fragment of CTL. First, this is a fragment that suffices to define all remaining operators in CTL. Second, in this fragment the semantics of both **EU** and **AF** operators can be represented as least fixed points of corresponding monotone functions. ALFP is a type of least fixed point logic and it is more convenient to specify least fixed points.

From Section 2.3.3, we know that characterizing the semantics of the **EG** operator amounts to calculating the greatest fixed point of a corresponding monotone function. We shall be careful if we want to specify **EG** in ALFP.

Assume that R_ϕ is a relation which specifies the set of states in a Kripke structure M that satisfy the formula ϕ and T is a binary relation which characterizes the transition relation of M. Let $R_{\mathbf{EG}\phi}$ be a relation which intends to specify the set of states (in M) that satisfy the formula $\mathbf{EG}\phi$. We cannot define the relation $R_{\mathbf{EG}\phi}$ using the following clause

$$\forall s : [R_\phi(s) \wedge \exists s' : T(s, s') \wedge R_{\mathbf{EG}\phi}(s)] \Rightarrow R_{\mathbf{EG}\phi}(s),$$

since calculating the least model of the above clause amounts to calculating the least fixed point of the function

$$F_{\mathbf{EG}\phi}(X) \equiv [\![\phi]\!] \cap \{s | \exists s' : s \to s' \ \wedge s' \in X\}.$$

However, $[\![\mathbf{EG}\phi]\!]$ is the greatest fixed point of this function.

In the following, we start to explain our encoding. We first encode a Kripke structure $M = (S, T, L)$ into ALFP by defining corresponding relations in ϱ_0 as follows. Assume that the universe is $\mathcal{U} = S$,

- for each atomic proposition p we define a predicate P_p such that $\varrho_0(P_p)(s)$ if and only if $p \in L(s)$, and
- we define a binary relation T such that $\varrho_0(T)(s,t)$ if and only if $(s,t) \in T$.

Much as the way to solve CTL model checking problem introduced in Section 2.3.2, our flow logic approach also proceeds along the syntax directed way. For each formula ϕ we define a corresponding relation R_ϕ characterizing those states which satisfy the formula ϕ. For each syntactic category ϕ in CTL, we define a judgement of the form $\vec{R} \vdash \phi$. The ALFP clauses defining the judgement impose the constraints between the relation R_ϕ and other relevant relations or judgements. They encode the CTL semantics in an inductive way. The intention is that $(M, s) \models \phi$ holds whenever $\varrho(R_\phi)(s)$ holds in the least model ϱ satisfying $\vec{R} \vdash \phi \wedge \varrho_0 \subseteq \varrho$.

For the several occurrences of the same subformula in ϕ, we decompose them to their constituent subformulas in the same way. This guarantees that all occurrences of the same subformula are handled in the same way in the flow logic. The definition is given in Table 3.1.

The relation R_{true} corresponds to CTL formula **true** and therefore we use the ALFP clause $\forall s : R_{true}(s)$ to define it. For atomic proposition p, we need to use predefined relation P_p. The clause $\forall s : P_p(s) \Rightarrow R_p(s)$ makes sure that if $P_p(s)$ holds, then $R_p(s)$ also holds. For $\neg\phi$, the conjunct $\vec{R} \vdash \phi$ ensures that the relation R_ϕ correctly records the set of states which satisfy ϕ. The clause $\forall s : \neg R_\phi(s) \Rightarrow R_{\neg\phi}(s)$ makes sure that if $R_\phi(s)$ does not hold, then $R_{\neg\phi}(s)$.

For $\phi_1 \vee \phi_2$, the conjuncts $\vec{R} \vdash \phi_1$ and $\vec{R} \vdash \phi_2$ ensures that the relation R_{ϕ_i} correctly approximate the set of states which satisfy ϕ_i $(i = 1, 2)$. The third

$$
\begin{array}{lll}
\vec{R} \vdash \mathbf{true} & \underline{\texttt{iff}} & [\forall s : R_{true}(s)] \\
\vec{R} \vdash p & \underline{\texttt{iff}} & [\forall s : P_p(s) \Rightarrow R_p(s)] \\
\vec{R} \vdash \phi_1 \vee \phi_2 & \underline{\texttt{iff}} & \vec{R} \vdash \phi_1 \wedge \vec{R} \vdash \phi_2 \wedge \\
 & & [\forall s : R_{\phi_1}(s) \vee R_{\phi_2}(s) \Rightarrow R_{\phi_1 \vee \phi_2}(s)] \\
\vec{R} \vdash \neg\phi & \underline{\texttt{iff}} & \vec{R} \vdash \phi \wedge \\
 & & [\forall s : \neg R_{\phi}(s) \Rightarrow R_{\neg\phi}(s)] \\
\vec{R} \vdash \mathbf{EX}\phi & \underline{\texttt{iff}} & \vec{R} \vdash \phi \wedge \\
 & & [\forall s : [\exists s' : T(s,s') \wedge R_{\phi}(s')] \Rightarrow R_{\mathbf{EX}\phi}(s)] \\
\vec{R} \vdash \mathbf{E}[\phi_1 \mathbf{U} \phi_2] & \underline{\texttt{iff}} & \vec{R} \vdash \phi_1 \wedge \vec{R} \vdash \phi_2 \wedge \\
 & & [\forall s : R_{\phi_2}(s) \Rightarrow R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}(s)] \wedge \\
 & & [\forall s : [\exists s' : T(s,s') \wedge R_{\phi_1}(s) \wedge R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}(s')] \\
 & & \Rightarrow R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}(s)] \\
\vec{R} \vdash \mathbf{AF}\phi & \underline{\texttt{iff}} & \vec{R} \vdash \phi \wedge \\
 & & [\forall s : R_{\phi}(s) \Rightarrow R_{\mathbf{AF}\phi}(s)] \wedge \\
 & & [\forall s : [\forall s' : \neg T(s,s') \vee R_{\mathbf{AF}\phi}(s')] \Rightarrow R_{\mathbf{AF}\phi}(s)]
\end{array}
$$

Table 3.1: CTL in ALFP

part caters for the relation $R_{\phi_1 \vee \phi_2}$ and captures the semantics of disjunction.

For $\mathbf{EX}\phi$, the first conjunct ensures that the subformula ϕ is handled correctly. The second conjunct tells that for any state s, if there exists a state s' such that both $T(s,s')$ and $R_{\phi}(s')$ hold, then $R_{\mathbf{EX}\phi}(s)$ also holds. This is exactly what the semantics for the **EX** operator tells us.

The clause for $\mathbf{E}[\phi_1 \mathbf{U} \phi_2]$ also captures the semantics for the **EU** operator and is based on the fact that $[\![\mathbf{E}[\phi_1 \mathbf{U} \phi_2]]\!]$ is the least fixed point of the function

$$F_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}(X) = [\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \{s | \exists s' : s \to s' \wedge s' \in X\}).$$

For any state s such that $R_{\phi_2}(s)$ holds, we require that $R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}(s)$ also holds. Alternatively if $R_{\phi_1}(s)$ holds and there exists a state s' such that both $T(s,s')$ and $R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}(s')$ hold, then $R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}(s)$ also holds.

The clause for $\mathbf{AF}\phi$ is also quite straightforward and is based on the fact that $[\![\mathbf{AF}\phi]\!]$ is the least fixed point of the function

$$F_{\mathbf{AF}\phi}(X) = [\![\phi]\!] \cup \{s | \forall s' : s \to s' \texttt{ implies } s' \in X\}.$$

For any state s such that $R_{\phi}(s)$ holds, $R_{\mathbf{AF}\phi}(s)$ also holds. Alternatively for states s and s', if $T(s, s')$ implies $R_{\mathbf{AF}\phi}(s')$, then $R_{\mathbf{AF}\phi}(s)$ also holds.

Ranking of ALFP formulas: It is easy to see that clauses for $\vec{R} \vdash \phi$ defined in Table 3.1 are indeed *closed* for any CTL formula ϕ. To show that the clauses are *stratified* we shall introduce our ranking method for the ALFP clauses defining the judgements in Table 3.1. We introduce the definition of *depth* of CTL formulas first. The *depth* of CTL formulas is defined as follows:

$$
\begin{array}{lll}
depth(\mathbf{true}) & = & 0 \\
depth(p) & = & 0 \\
depth(\neg\phi) & = & 1 + depth(\phi) \\
depth(\phi_1 \vee \phi_2) & = & 1 + max\{depth(\phi_1), depth(\phi_2)\} \\
depth(\mathbf{EX}\phi) & = & 1 + depth(\phi) \\
depth(\mathbf{E}[\phi_1 \mathbf{U} \phi_2]) & = & 1 + max\{depth(\phi_1), depth(\phi_2)\} \\
depth(\mathbf{AF}\phi) & = & 1 + depth(\phi)
\end{array}
$$

We assign the transition relation T and the relation P_p the rank 0. For each CTL formula ϕ and the corresponding relation R_ϕ, we require that the rank of the relation R_ϕ equals to the *depth* of the formula ϕ. That is

$$rank_{R_\phi} = depth(\phi)$$

Example 3.1 *Let* $\phi = \boldsymbol{E}[(p_1 \vee \neg p_2)\, \boldsymbol{U} p_3]$ *be a CTL formula. According to our ranking method, we require that* $rank_{R_{p_1}} = 0$, $rank_{R_{p_2}} = 0$, $rank_{R_{\neg p_2}} = 1$, $rank_{R_{p_1 \vee \neg p_2}} = 1$, $rank_{R_{p_3}} = 0$, *and* $rank_{R_{\boldsymbol{E}[(p_1 \vee \neg p_2)\boldsymbol{U} p_3]}} = 1$. *It is easy to see that the clauses for* $\vec{R} \vdash \boldsymbol{E}[(p_1 \vee \neg p_2)\, \boldsymbol{U} p_3]$ *is stratified.*

We have the following lemma which guarantees stratification for all clauses generated for judgements defined in Table 3.1.

Lemma 3.1 *The ALFP clauses generated for judgements* $\vec{R} \vdash \phi$ *defined in Table 3.1 are closed and stratified.*

Proof. In Appendix A. □

Example 3.2 *Let's go back to Example 2.3 and show how to use flow logic approach to solve the same problem. For each subformula of* $\boldsymbol{AF}(p \vee q)$, *we define a relation for it to record the set of states which satisfy the subformula. We use the judgement* $\vec{R} \vdash \boldsymbol{AF}(p \vee q)$ *to specify our static analysis for* $\boldsymbol{AF}(p \vee q)$.

The judgement will generate new judgements and ALFP clauses, according to Table 3.1, listed as follows:

$$\begin{array}{l} \vec{R} \vdash (p \vee q) \wedge \\ [\forall s : R_{(p\vee q)}(s) \Rightarrow R_{\boldsymbol{AF}(p\vee q)}(s)] \wedge \\ [\forall s : [\forall s' : \neg T(s,s') \vee R_{\boldsymbol{AF}(p\vee q)}(s')] \Rightarrow R_{\boldsymbol{AF}(p\vee q)}(s)] \end{array}$$

We can also generate new clauses for the newly created judgement $\vec{R} \vdash (p \vee q)$, and the process will continue until we generate all the clauses. Since our flow logic table is syntax directed and well-defined, the process will terminate and we list all the clauses we get in the example as follows:

$$\begin{array}{l} [\forall s : P_p(s) \Rightarrow R_p(s)] \wedge \\ [\forall s : P_q(s) \Rightarrow R_q(s)] \wedge \\ [\forall s : R_p(s) \vee R_q(s) \Rightarrow R_{(p\vee q)}(s)] \wedge \\ [\forall s : R_{(p\vee q)}(s) \Rightarrow R_{\boldsymbol{AF}(p\vee q)}(s)] \wedge \\ [\forall s : [\forall s' : \neg T(s,s') \vee R_{\boldsymbol{AF}(p\vee q)}(s')] \Rightarrow R_{\boldsymbol{AF}(p\vee q)}(s)] \end{array}$$

According to the Moore family property, there exists a best analysis result in our example, that is the least model for the above generated ALFP clauses. The interpretation for each relation in the least model is listed below and if we compare it with the solution for CTL model checking given in Example 2.3, we will see that they are exactly the same.

R_ϕ	$\{s \mid R_\phi(s)\}$
R_p	$\{s_1\}$
R_q	$\{s_2\}$
$R_{(p\vee q)}$	$\{s_1, s_2\}$
$R_{\boldsymbol{AF}(p\vee q)}$	$\{s_1, s_2, s_3\}$

In the following, we introduce our main theorem in this section.

Theorem 3.2 *Given a CTL formula ϕ and an initial interpretation ϱ_0 which defines T and P_p. Assume that ϱ is the least solution to $\vec{R} \vdash \phi \;\wedge \varrho \supseteq \varrho_0$, we have $(M, s) \models \phi$ iff $s \in \varrho(R_\phi)$.*

PROOF. In Appendix A. □

Continuing the discussion in the beginning of this section, let's look at how to specify the **EG** operator in ALFP. We can define the relation $R_{\mathbf{EG}\phi}$ as follows which is based on the equivalence $\mathbf{EG}\phi \equiv \neg\mathbf{AF}\neg\phi$:

$$\begin{array}{lll} \vec{R} \vdash \mathbf{EG}\phi & \underline{\mathtt{iff}} & \vec{R} \vdash \mathbf{AF}\neg\phi \wedge \\ & & [\forall s : \neg R_{\mathbf{AF}\neg\phi}(s) \Rightarrow R_{\mathbf{EG}\phi}(s)] \end{array}$$

Stratification is guaranteed in the above specification. In the next section, we will show another way of specifying the **EG** operator in ALFP. There, the fairness problems in CTL is also taken into consideration.

3.2 CTL with Fairness Constraints in ALFP

In this section, we show how to use flow logic to encode CTL with fairness assumptions in ALFP. Here, we express CTL formulas in Existential Normal Form given in Definition 2.7.

Table 2.4 in Section 2.3.4 has introduced properties for formulas of the forms $\mathbf{EX}p$, $\mathbf{E}[p_1\mathbf{U}p_2]$ and $\mathbf{EG}p$, where p, p_1 and p_2 are atomic propositions, under fairness assumptions. These insights are very useful when developing model checking algorithms [10]. More generally, to consider formulas of the forms $\mathbf{EX}\phi$, $\mathbf{E}[\phi_1\mathbf{U}\phi_2]$ and $\mathbf{EG}\phi$, we need to modify Table 2.4 properly. We give the following properties for the **EX**, **EU** and **EG** operators under fairness assumptions in Table 3.2. This also helps to give semantic insights into these operators.

It is very easy to verify the correctness of Table 3.2. Let's take the **EX** operator as an example to explain the difference between Table 2.4 and Table 3.2. In Table 2.4, we know that the following holds:

$$\begin{array}{lll} (M,s) \models_{fair} \mathbf{EX}p & \underline{\mathtt{iff}} & \texttt{there exists a state } s' \texttt{ such that } s \rightarrow s', \\ & & (M,s') \models p \texttt{ and } Path_{fair}^{true}(s') \neq \emptyset \end{array}$$

There, we only require $(M,s') \models p$ instead of $(M,s') \models_{fair} p$. This simplification is due to the fact that $(M,s) \models p$ iff $(M,s) \models_{fair} p$. However, when we consider the semantics of $\mathbf{EX}\phi$ in Table 3.2, we must require that $(M,s') \models_{fair} \phi$.

$(M,s) \models_{fair} \mathbf{EX}\phi$	iff	there exists a state s' such that $s \rightarrow s'$, $(M,s') \models_{fair} \phi$ and $Path^{true}_{fair}(s') \neq \emptyset$
$(M,s) \models_{fair} \mathbf{E}[\phi_1 \mathbf{U} \phi_2]$	iff	there exists a finite path fragment $\pi_{fin} = s_0, ..., s_k$ such that $s = s_0$, $(M,s_k) \models_{fair} \phi_2, \forall\, 0 \leq j < k : (M,s_j) \models_{fair} \phi_1$, and $Path^{true}_{fair}(s_k) \neq \emptyset$
$(M,s) \models_{fair} \mathbf{EG}\phi$	iff	$Path^{\phi}_{fair}(s) \neq \emptyset$

Table 3.2: Properties for **EX**, **EU** and **EG** operators under fairness assumptions

Otherwise, this would be inconsistent with the semantics of CTL with fairness assumptions. The difference for the case of the **EU** operator is similar.

In the following, we shall develop an ALFP-based static analysis technique to encode CTL with fairness assumptions. We first introduce a notation $PATH_{fair,\mathcal{S}}$ in the following.

Definition 3.3 Given a Kripke structure $M = (S, T, L)$ and a fixed CTL fairness assumption `fair`, we define that $PATH_{fair,\mathcal{S}} = \{s | \exists \pi : \pi[0] = s \wedge \pi \models fair \wedge \forall 0 \leq i : \pi[i] \in \mathcal{S}\}$ where $\mathcal{S} \subseteq S$.

Recall that we have defined that $Path^{\phi}_{fair}(s) = \{\pi |\ \pi$ is a path in M_ϕ and $\pi \models fair \wedge \pi[0] = s\}$ in Section 2.3.4. Following lemma shows the relation between $PATH_{fair,\mathcal{S}}$ and $Path^{\phi}_{fair}(s)$.

Lemma 3.4 *Assume that $\varrho(R_\phi) = \{s | (M,s) \models_{fair} \phi\}$. We have that $PATH_{fair,\varrho(R_\phi)} = \{s | \exists \pi : \pi \in Path^{\phi}_{fair}(s)\}$.*

Proof. It follows directly from the definition of M_ϕ and $Path^{\phi}_{fair}(s)$ and Definition 3.3. □

To characterize the semantics of CTL with fairness assumptions, we also need to define an extra relation $Path_{fair,\phi}$ to approximate the set of states from which there exists fair paths in M_ϕ. We introduce the notation $\vec{R} \Vdash Path_{fair,\phi}$ in the following to denote a set of ALFP clauses which define $Path_{fair,\phi}$. The details of $\vec{R} \Vdash Path_{fair,\phi}$ are postponed to the following sections, where we will define $\vec{R} \Vdash Path_{ufair,\phi}$ and $\vec{R} \Vdash Path_{sfair,\phi}$ corresponding to unconditional fairness and strong fairness respectively. Weak fairness can be considered as a special case of unconditional fairness problems.

Definition 3.5 Given a fixed CTL fairness assumption `fair` and a CTL formula ϕ, we use $\vec{R} \Vdash Path_{fair,\phi}$ to denote a set of ALFP clauses which define the relation $Path_{fair,\phi}$.

We encode a Kripke structure $M = (S, T, L)$ into ALFP by defining corresponding relations in ϱ_0 in the same way as we have done in the previous section. We give our flow logic approach to the analysis of CTL with fairness assumptions in Table 3.3. There, $Path_{fair,\phi}(s)$ (resp. $Path_{fair,true}(s)$) intends to mean that $Path^{\phi}_{fair}(s) \neq \emptyset$ (resp. $Path^{true}_{fair}(s) \neq \emptyset$). Our analysis for **EX**, **EU** and **EG** operators are based on the properties of these operators listed in Table 3.2.

$\vec{R} \vdash_{fair} \mathbf{true}$	<u>`iff`</u>	$[\forall s : R_{true}(s)]$
$\vec{R} \vdash_{fair} p$	<u>`iff`</u>	$[\forall s : P_p(s) \Rightarrow R_p(s)]$
$\vec{R} \vdash_{fair} \neg\phi$	<u>`iff`</u>	$\vec{R} \vdash_{fair} \phi \wedge$ $[\forall s : (\neg R_{\phi}(s)) \Rightarrow R_{\neg\phi}(s)]$
$\vec{R} \vdash_{fair} \phi_1 \wedge \phi_2$	<u>`iff`</u>	$\vec{R} \vdash_{fair} \phi_1 \wedge \vec{R} \vdash_{fair} \phi_2 \wedge$ $[\forall s : R_{\phi_1}(s) \wedge R_{\phi_2}(s) \Rightarrow R_{\phi_1 \wedge \phi_2}(s)]$
$\vec{R} \vdash_{fair} \mathbf{EX}\phi$	<u>`iff`</u>	$\vec{R} \vdash_{fair} \phi \wedge \vec{R} \Vdash Path_{fair,true}$ $[\forall s : [\exists s' : T(s,s') \wedge R_{\phi}(s') \wedge Path_{fair,true}(s')]$ $\Rightarrow R_{\mathbf{EX}\phi}(s)]$
$\vec{R} \vdash_{fair} \mathbf{E}[\phi_1 \mathbf{U} \phi_2]$	<u>`iff`</u>	$\vec{R} \vdash_{fair} \phi_1 \wedge \vec{R} \vdash_{fair} \phi_2 \wedge \vec{R} \Vdash Path_{fair,true}$ $[\forall s : R_{\phi_2}(s) \wedge Path_{fair,true}(s) \Rightarrow R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}(s)] \wedge$ $[\forall s : [\exists s' : T(s,s') \wedge R_{\phi_1}(s) \wedge R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}(s')]$ $\Rightarrow R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}(s)]$
$\vec{R} \vdash_{fair} \mathbf{EG}\phi$	<u>`iff`</u>	$\vec{R} \vdash_{fair} \phi \wedge \vec{R} \Vdash Path_{fair,\phi}$ $[\forall s : Path_{fair,\phi}(s) \Rightarrow R_{\mathbf{EG}\phi}(s)]$

Table 3.3: CTL with Fairness Assumptions in ALFP

The encoding in Table 3.3 is not complicated. We explain briefly the case of $\mathbf{EX}\phi$ and $\mathbf{EG}\phi$ in the following.

For $\mathbf{EX}\phi$, the first conjunct ensures that we analyze the subformula ϕ correctly. The second conjunct guarantees that the relation $Path_{fair,true}$ is defined correctly. Here, $Path_{fair,true}$ intends to characterize the set of states from which there exist fair paths in M. The third conjunct tells that for any state s, if there exists a state s' such that both $T(s,s')$ and $R_\phi(s')$ hold and $Path_{fair,true}(s')$, then $R_{\mathbf{EX}\phi}(s)$ also holds. This matches the property of the $\mathbf{EX}$ operator given in Table 3.2.

For $\mathbf{EG}\phi$, the first conjunct also ensures that we handle the subformula ϕ as intended. The conjunct $\vec{R} \Vdash Path_{fair,\phi}$ ensures that the relation $Path_{fair,\phi}$, which intends to characterize the set of states from which there exist fair paths in M_ϕ, is defined correctly. The third conjunct tells that for any state s, if there exists a fair path in M_ϕ from s, then $R_{\mathbf{EG}\phi}(s)$ holds. As to how to define the relation $Path_{fair,\phi}$, we explain it in the next two sections in the setting of unconditional fairness and strong fairness. In both setting, we will be essentially calculating nontrivial strongly connected components in a Kripke structure M_ϕ.

Following is the main theorem of this section. There, we have made an assumption that $\varrho(Path_{fair,\varphi}) = PATH_{fair,\varrho(R_\varphi)}$. When $\varrho(R_\phi) = \{s|(M,s) \models_{fair} \phi\}$ holds, we know from Lemma 3.4 that $\varrho(Path_{fair,\varphi}) = \{s|\exists\pi : \pi \in Path^{\phi}_{fair}(s)\}$.

Theorem 3.6 *Given a CTL formula ϕ, a fixed CTL fairness assumption $\mathtt{fair}$, and an initial interpretation ϱ_0 which defines T and P_p. Assume that ϱ is the least solution to $\vec{R} \vdash_{fair} \phi \ \wedge \varrho \supseteq \varrho_0$ and that $\varrho(Path_{fair,\varphi}) = PATH_{fair,\varrho(R_\varphi)}$ whenever φ is $\mathtt{true}$ or a subformula of ϕ, we have that $(M,s) \models_{fair} \phi$ iff $s \in \varrho(R_\phi)$.*

Proof. In Appendix A. □

3.2.1 Unconditional Fairness and Weak Fairness

In this section, we discus unconditional fairness and weak fairness problems. We will first show how to define $\vec{R} \Vdash Path_{ufair,\phi}$, which is a set of ALFP clauses that define the relation $Path_{ufair,\phi}$. Recall that $Path_{ufair,\phi}$ intends to characterize the set of states (in M_ϕ) from which there exist unconditional fair paths in M_ϕ. Then we point out that weak fairness is a special case of unconditional fairness so that our result for unconditional fairness applies to weak fairness as well.

Calculating nontrivial strongly connected set plays an important role here as well as in the next section. We define it as follows.

Definition 3.7 Let M be a finite state Kripke structure. A set of states C in M is strongly connected if for any pair of states s and s' in C there is a finite path fragment $\pi_{fin} = s_0, s_1 ... s_n (n \geq 0, \forall 0 \leq i \leq n : s_i \in C)$ in M such that $s_0 = s$ and $s_n = s'$.

A strongly connected set C is *trivial* if C only contains one state s and there is no self-loop on s. In the rest of the thesis, we are mainly interested in calculating nontrivial strongly connected sets. The following fact gives the relationship between nontrivial strongly connected sets and infinite paths in a finite state Kripke structure.

Fact 3.2.1 *Let M be a finite state Kripke structure. There is an infinite path from a state s iff there exists a nontrivial strongly connected set C in M such that C is reachable from s.*

We explain Fact 3.2.1 briefly. We first explain one direction. Assume that M has n states. Let $\pi = s_0, s_1 ...$ be an infinite path in M such that $s_0 = s$. In the prefix $\pi_{fin} = s_0, ..., s_n$ of π, we know that at least one state has been visited twice since there are only n states in M. Assume that s' has been visited twice in π_{fin}. Then, the finite path fragment $\pi'_{fin} = s_i, ..., s_j$ in π_{fin} such that $s_i = s_j = s'$ $(0 \leq i, j \leq n)$ forms a cycle. We can thus construct a nontrivial strongly connected set $C = \{s | s \text{ occurs in } \pi'_{fin}\}$ that is reachable from s.

The other direction is obvious. Assume that a nontrivial strongly connected set C is reachable from s. Then, there exists a finite path fragment $\pi_{fin} = s_0, ..., s_i$ such that $s_0 = s$ and $s_i \in C$ $(0 \leq i)$. We can easily extend π_{fin} to an infinite path. Assume that C has more than one state. In this case, let s_j $(0 \leq j)$ be a state in C such that $s_i \neq s_j$. We can find a finite path fragment π'_{fin} from s_i to s_j and another path fragment π''_{fin} from s_j to s_i. The two path fragments form a cycle. Therefore, starting from s we could first go to s_i and then go back and forth between s_i and s_j infinitely many times. This forms an infinite path from s. Assume that C has only one state. Since C is nontrivial. s_i has a self-loop. Therefore, starting from s we could first go to s_i and then self loop on state s_i.

This forms an infinite path from s as well.

Since we have required that a Kripke structure M is total, from a state s there is always an infinite path in M. However, it is not guaranteed that a Kripke structure M_ϕ is total. Fact 3.2.1 is useful when we want to know whether there is an infinite path from a state s in M_ϕ.

Unconditional fairness constraints have the form $ufair = \bigwedge_{1\le i\le k} \mathbf{GF}b_i$, where b_i is an atomic proposition. We focus on a Kripke structure M_ϕ. Due to the following lemma, we say that the constraint $\mathbf{GF}b_i$ is realizable in a nontrivial strongly connected set C if $C \cap \{s|(M_\phi, s) \models b_i\} \neq \emptyset$.

Lemma 3.8 *Assume that $ufair = \bigwedge_{1\le i\le k} \boldsymbol{GF}b_i$. Let C be a nontrivial strongly connected set in M_ϕ such that $C \cap \{s|(M_\phi, s) \models b_i\} \neq \emptyset$ for all $1 \le i \le k$. For each state $s \in C$, there exists a path π in M_ϕ from s such that $\pi \models ufair$.*

Proof. In Appendix A. □

Let $uFairSCSs_\phi$ denote the set union of all nontrivial strongly connected sets C in M_ϕ such that $C \cap \{s|(M_\phi, s) \models b_i\} \neq \emptyset$ for all $1 \le i \le k$. Fact 2.3.2 and 3.2.1 and Lemma 3.8 lead to the following lemma.

Lemma 3.9 *Assume that $ufair = \bigwedge_{1\le i\le k} \boldsymbol{GF}b_i$. There exists an unconditional fair path in M_ϕ from s if and only if there exists a finite path fragment π_{fin} (in M_ϕ) from s to a state s' in $uFairSCSs_\phi$.*

Proof. In Appendix A. □

Based on Lemma 3.9, we define $\vec{R} \Vdash Path_{ufair,\phi}$ as follows, where we define relations T_ϕ, T_ϕ^+, SC_ϕ, $SC_{ufair,\phi}$ and $Path_{ufair,\phi}$:

$$[\forall s : \forall s' : [T(s, s') \wedge R_\phi(s) \wedge R_\phi(s') \Rightarrow T_\phi(s, s')]]$$
$$[\forall s : \forall s' : [T_\phi(s, s') \Rightarrow T_\phi^+(s, s')]]$$
$$[\forall s : \forall s'' : [\exists s' : T_\phi^+(s, s') \wedge T_\phi^+(s', s'') \Rightarrow T_\phi^+(s, s'')]]$$
$$[\forall s : \forall s' : [T_\phi^+(s, s') \wedge T_\phi^+(s', s) \Rightarrow SC_\phi(s, s')]]$$

$$[\forall s : \bigwedge_{1\leq i\leq k} [\exists s_i : SC_\phi(s, s_i) \wedge P_{b_i}(s_i)] \Rightarrow SC_{ufair,\phi}(s)]$$

$$[\forall s : [\exists s' : T^+_\phi(s, s') \wedge SC_{ufair,\phi}(s')] \Rightarrow Path_{ufair,\phi}(s)]$$

The following lemma shows the correctness of our definition for $\vec{R} \Vdash Path_{ufair,\phi}$.

Lemma 3.10 *Let ϱ_0 be an initial interpretation which defines T, P_p and R_ϕ. Assume that $\varrho_0(R_\phi) = \{s|(M, s) \models_{ufair} \phi\}$. For the least solution ϱ to $\vec{R} \Vdash Path_{ufair,\phi} \wedge \varrho \supseteq \varrho_0$, we have the following:*

- *$\varrho(T_\phi)$ equals the transition relation in M_ϕ,*
- *$(s, s') \in \varrho(T^+_\phi)$ iff there exists a finite path fragment $\pi_{fin} = s_0, s_1...s_n$ in M_ϕ where $s_0 = s$ and $s_n = s'$,*
- *$(s, s') \in \varrho(SC_\phi)$ iff s and s' belong to a nontrivial strongly connected set in M_ϕ,*
- *$\varrho(SC_{ufair,\phi}) = uFairSCSs_\phi$, and*
- *$\varrho(Path_{ufair,\phi}) = \{s|\exists\pi : \pi \in Path^\phi_{ufair}(s)\}$.*

Proof. In Appendix A. □

The following corollary shows that when fairness assumptions take the form of *ufair*, the assumption that $\varrho(Path_{ufair,\phi}) = PATH_{ufair,\varrho_0(R_\phi)}$, which is made in Theorem 3.6, holds.

Corollary 3.11 *Let ϱ_0 be an initial interpretation which defines T, P_p and R_ϕ. Assume that $\varrho_0(R_\phi) = \{s|(M, s) \models_{ufair} \phi\}$. The least solution ϱ to $\vec{R} \Vdash Path_{ufair,\phi} \wedge \varrho \supseteq \varrho_0$ satisfies $\varrho(Path_{ufair,\phi}) = PATH_{ufair,\varrho_0(R_\phi)}$.*

Proof. It's obvious from Lemma 3.4 and Lemma 3.10. □

Weak fairness constraints take the form $wfair = \bigwedge_{1\leq i\leq k}(\mathbf{FG}a_i \Rightarrow \mathbf{GF}b_i)$. The following equivalences hold for weak fairness constraints:

$$
\begin{aligned}
wfair = \bigwedge_{1\le i\le k} (\mathbf{FG}a_i \Rightarrow \mathbf{GF}b_i) &\equiv \bigwedge_{1\le i\le k} \neg\mathbf{FG}a_i \vee \mathbf{GF}b_i \\
&\equiv \bigwedge_{1\le i\le k} \mathbf{G}\neg\mathbf{G}a_i \vee \mathbf{GF}b_i \\
&\equiv \bigwedge_{1\le i\le k} \mathbf{GF}\neg a_i \vee \mathbf{GF}b_i \\
&\equiv \bigwedge_{1\le i\le k} \mathbf{GF}(\neg a_i \vee b_i)
\end{aligned}
$$

Therefore, weak fairness constraints is a special case of unconditional fairness and can be encoded in ALFP in a similar way.

Remark: In this section, we have actually provided another way to specify the semantics of the **EG** operator (without fairness assumptions) in comparison with the way we developed in the previous section. This is done by taking a special form of unconditional fairness constraints $ufair = \mathbf{GFtrue}$. Since it is obvious that **true** always holds, each infinite path in a Kripke structure is an unconditional fair path. Therefore, we know that $(M, s) \models_{ufair} \phi$ iff $(M, s) \models \phi$. In this case, for $M_\phi = (S, T_\phi, L)$, we know that $(s, s') \in T_\phi$ if and only if $(s, s') \in T$ and $(M, s) \models \phi$ and $(M, s') \models \phi$.

The clauses we have used to define $\vec{R} \Vdash Path_{ufair,\phi}$ specialize to the following:

$$
\begin{gathered}
[\forall s : \forall s' : [T(s, s') \wedge R_\phi(s) \wedge R_\phi(s') \Rightarrow T_\phi(s, s')]] \\
[\forall s : \forall s' : [T_\phi(s, s') \Rightarrow T^+_\phi(s, s')]] \\
[\forall s : \forall s'' : [\exists s' : T^+_\phi(s, s') \wedge T^+_\phi(s', s'') \Rightarrow T^+_\phi(s, s'')]] \\
[\forall s : \forall s' : [T^+_\phi(s, s') \wedge T^+_\phi(s', s) \Rightarrow SC_\phi(s, s')]] \\
[\forall s : \exists s' : SC_\phi(s, s') \Rightarrow SC_{ufair,\phi}(s)] \\
[\forall s : [\exists s' : T^+_\phi(s, s') \wedge SC_{ufair,\phi}(s')] \Rightarrow Path_{ufair,\phi}(s)]
\end{gathered}
$$

From Lemma 3.10, for the least solution ϱ to the above clauses, we know that $\varrho(SC_{ufair,\phi})$ equals to the set union of all nontrivial strongly connected set in M_ϕ and $\varrho(Path_{ufair,\phi}) = \{s|$ there exists an infinite path from s in M_ϕ $\}$. From Fact 2.3.1, we know that the semantics of the **EG** operator (without fairness) can be specified by $[\forall s : Path_{ufair,\phi}(s) \Rightarrow R_{\mathbf{EG}\phi}(s)]$, where $ufair = \mathbf{GFtrue}$.

Actually, it has been pointed out in [2] that $(M, s) \models \mathbf{EG}\phi$ iff there exists a path in M_ϕ that leads from the state s to some state s' in a nontrivial strongly connected component in M_ϕ. This provides some semantic insights into our encoding.

3.2.2 Strong Fairness

In this section, we consider how to define $\vec{R} \Vdash Path_{sfair,\phi}$, which denotes a set of ALFP clauses that define the relation $Path_{sfair,\phi}$. Recall that $Path_{sfair,\phi}$ intends to characterize the set of states (in M_ϕ) from which there exist strong fair paths in M_ϕ.

Strong fairness constraints have the form $sfair = \bigwedge_{1\leq i\leq k}(\mathbf{GF}a_i \Rightarrow \mathbf{GF}b_i)$, where a_i and b_i are atomic propositions. Let us first consider the case $k = 1$ and now $sfair$ has the form $sfair = \mathbf{GF}a \Rightarrow \mathbf{GF}b$.

The constraint $sfair$ is realizable in a nontrivial strongly connected set C if either $C \cap \{s|(M_\phi, s) \models b\} \neq \emptyset$ or $\forall s \in C : \{s|(M_\phi, s) \not\models a\}$ holds. The condition $C \cap \{s|(M_\phi, s) \models b\} \neq \emptyset$ ensures that from each state in C, there exists an infinite path π on which the proposition b is satisfied infinitely often. Therefore, $\mathbf{GF}b$ is satisfied on π, which means π is a strong fair path. The other condition $\forall s \in C : \{s|(M_\phi, s) \not\models a\}$ makes sure that there is no state in C that satisfies the proposition a. In this case, on any infinite path π starting from a state in C, we know that a is not satisfied infinitely often, which means $\mathbf{GF}a$ is not satisfied on π. This also means that π is a strong fair path. We formalize this observation in the following lemma.

Lemma 3.12 *Assume that $sfair = \mathbf{GF}a \Rightarrow \mathbf{GF}b$. Let C be a nontrivial strongly connected set in M_ϕ such that either $C \cap \{s|(M_\phi, s) \models b\} \neq \emptyset$ or $\forall s \in C : \{s|(M_\phi, s) \not\models a\}$. For each state $s \in C$, there exists a path π in M_ϕ from s such that $\pi \models sfair$.*

Proof. In Appendix A. $\square$

Let $sFairSCSs_\phi$ denote the set union of all nontrivial strongly connected sets C in M_ϕ which satisfy either $C \cap \{s|(M_\phi, s) \models b\} \neq \emptyset$ or $\forall s \in C : \{s|(M_\phi, s) \not\models a\}$. Fact 2.3.2 and 3.2.1 and Lemma 3.12 lead to the following lemma.

Lemma 3.13 *Assume that $sfair = \boldsymbol{GF}a \Rightarrow \boldsymbol{GF}b$. There exists a strong fair path in M_ϕ from s if and only if there exists a finite path fragment π_{fin}, in M_ϕ, from s to a state s' in $sFairSCSs_\phi$.*

Proof. In Appendix A. □

Based on Lemma 3.13, we define $\vec{R} \Vdash Path_{sfair,\phi}$ in the following:

$$[\forall s : \forall s' : [T(s,s') \wedge R_\phi(s) \wedge R_\phi(s') \Rightarrow T_\phi(s,s')]]$$
$$[\forall s : \forall s' : [T_\phi(s,s') \Rightarrow T_\phi^+(s,s')]]$$
$$[\forall s : \forall s' : \forall s'' : [T_\phi^+(s,s') \wedge T_\phi^+(s',s'') \Rightarrow T_\phi^+(s,s'')]]$$
$$[\forall s : \forall s' : [T_\phi^+(s,s') \wedge T_\phi^+(s',s) \Rightarrow SC_\phi(s,s')]]$$
$$[\forall s : \forall s' : [T_\phi(s,s') \wedge R_{\neg a}(s) \wedge R_{\neg a}(s') \Rightarrow T_{\phi\wedge\neg a}(s,s')]]$$
$$[\forall s : \forall s' : [T_{\phi\wedge\neg a}(s,s') \Rightarrow T_{\phi\wedge\neg a}^+(s,s')]]$$
$$[\forall s : \forall s' : \forall s'' : [T_{\phi\wedge\neg a}^+(s,s') \wedge T_{\phi\wedge\neg a}^+(s',s'') \Rightarrow T_{\phi\wedge\neg a}^+(s,s'')]]$$
$$[\forall s : \forall s' : [T_{\phi\wedge\neg a}^+(s,s') \wedge T_{\phi\wedge\neg a}^+(s',s) \Rightarrow SC_{\phi\wedge\neg a}(s,s')]]$$
$$[\forall s : [\exists s' : SC_\phi(s,s') \wedge R_b(s')] \Rightarrow SC_{sfair,\phi}(s)]$$
$$[\forall s : [\exists s' : SC_{\phi\wedge\neg a}(s,s')] \Rightarrow SC_{sfair,\phi}(s)]$$
$$[\forall s : [\exists s' : T_\phi^+(s,s') \wedge SC_{sfair,\phi}(s')] \Rightarrow Path_{sfair,\phi}(s)]$$

The following lemma shows the correctness of our definition of $\vec{R} \Vdash Path_{sfair,\phi}$.

Lemma 3.14 *Let ϱ_0 be an initial interpretation which defines T, P_p, $R_{\neg a}$ and R_ϕ. Assume that $\varrho_0(R_\phi) = \{s|(M,s) \models_{sfair} \phi\}$ and $\varrho_0(R_{\neg a}) = \{s|(M,s) \not\models a\}$. For the least solution ϱ to $\vec{R} \Vdash Path_{sfair,\phi} \wedge \varrho \supseteq \varrho_0$, we have the following:*

- *$\varrho(T_\phi)$ (resp. $\varrho(T_{\phi\wedge\neg a})$) equals the transition relation in M_ϕ (resp. $M_{\phi\wedge\neg a}$),*

- *$(s, s') \in \varrho(T^+_\phi)$ (resp. $(s, s') \in \varrho(T^+_{\phi\wedge\neg a})$) iff there exists a finite path fragment $\pi_{fin} = s_0, s_1 ... s_n$ in M_ϕ (resp. $M_{\phi\wedge\neg a}$) where $s_0 = s$ and $s_n = s'$,*
- *$(s, s') \in \varrho(SC_\phi)$ (resp. $(s, s') \in \varrho(SC_{\phi\wedge\neg a})$) iff s and s' belong to a non-trivial strongly connected set in M_ϕ (resp. $M_{\phi\wedge\neg a}$),*
- *$\varrho(SC_{sfair,\phi}) = sFairSCSs_\phi$, and*
- *$\varrho(Path_{sfair,\phi}) = \{s|\exists\pi : \pi \in Path^\phi_{sfair}(s)\}$.*

Proof. In Appendix A. □

The following corollary shows that when fairness assumptions take the form of $sfair = \mathbf{GF}a \Rightarrow \mathbf{GF}b$, the assumption that $\varrho(Path_{sfair,\phi}) = PATH_{sfair,\varrho_0(R_\phi)}$, which is made in Theorem 3.6, holds.

Corollary 3.15 *Let ϱ_0 be an initial interpretation which defines T, P_p, $R_{\neg a}$ and R_ϕ. Assume that $\varrho_0(R_\phi) = \{s|(M, s) \models_{sfair} \phi\}$ and $\varrho_0(R_{\neg a}) = \{s|(M, s) \not\models a\}$. The least solution ϱ to $\vec{R} \Vdash Path_{sfair,\phi} \wedge \varrho \supseteq \varrho_0$ satisfies $\varrho(Path_{sfair,\phi}) = PATH_{sfair,\varrho_0(R_\phi)}$.*

Proof. It's obvious from Lemma 3.4 and Lemma 3.14. □

Let us now consider the case $k > 1$ and now $sfair$ has the form $sfair = \bigwedge_{1\le i\le k}(\mathbf{GF}a_i \Rightarrow \mathbf{GF}b_i)$. The constraint $sfair$ is realizable in a nontrivial strongly connected set C if for each constraint $\mathbf{GF}a_i \Rightarrow \mathbf{GF}b_i$ $(1 \le i \le k)$, either $C \cap \{s|(M_\phi, s) \models b_i\} \ne \emptyset$ or $\forall s \in C : \{s|(M_\phi, s) \not\models a_i\}$ holds. Notice that there are 2^k different combinations such that $sfair$ is realizable in C. This leads to an exponential blow up, which can be seen from the following equivalence as well.

Let $\mathbb{B} = \{0, 1\}$ and $e \in \mathbb{B}^k$. We use $e[i]$ to denote the i-th boolean value in e. We have the following equivalences for strong fairness constraints:

$$sfair = \bigwedge_{1\le i\le k}(\mathbf{GF}a_i \Rightarrow \mathbf{GF}b_i) \equiv \bigwedge_{1\le i\le k}(\mathbf{FG}\neg a_i \vee \mathbf{GF}b_i)$$

$$\equiv \bigvee_{e\in\mathbb{B}^k}((\bigwedge_{\substack{1\leq i\leq k\\ e[i]=0}} \mathbf{FG}\neg a_i) \wedge (\bigwedge_{\substack{1\leq j\leq k\\ e[j]=1}} \mathbf{GF}b_j))$$

$$\equiv \bigvee_{e\in\mathbb{B}^k}(\mathbf{FG}(\bigwedge_{\substack{1\leq i\leq k\\ e[i]=0}} \neg a_i) \wedge (\bigwedge_{\substack{1\leq j\leq k\\ e[j]=1}} \mathbf{GF}b_j))$$

We have the following corollary.

Corollary 3.16 *Assume that* $sfair = \bigwedge_{1\leq i\leq k}(\boldsymbol{GF}a_i \Rightarrow \boldsymbol{GF}b_i)$. *Let* $sFairSCSs_\phi$ *denote the set union of all nontrivial strongly connected sets* C *in* M_ϕ *such that for all* $1 \leq i \leq k$, C *satisfy either* $C \cap \{s|(M_\phi, s) \models b_i\} \neq \emptyset$ *or* $\forall s \in C : \{s|(M_\phi, s) \not\models a_i\}$. *There exists a strong fair path in* M_ϕ *from* s *if and only if there exists a finite path fragment* π_{fin} *(in* M_ϕ*) from* s *to a state* s' *in* $sFairSCSs_\phi$.

Proof. It is straightforward based on Lemma 3.13. □

We show how to define $\vec{R} \Vdash Path_{sfair,\phi}$ when $k > 1$. We define relations T_ϕ^+ and SC_ϕ in the following clauses:

$$[\forall s : \forall s' : [T(s, s') \wedge R_\phi(s) \wedge R_\phi(s') \Rightarrow T_\phi(s, s')]]$$
$$[\forall s : \forall s' : [T_\phi(s, s') \Rightarrow T_\phi^+(s, s')]]$$
$$[\forall s : \forall s' : \forall s'' : [T_\phi^+(s, s') \wedge T_\phi^+(s', s'') \Rightarrow T_\phi^+(s, s'')]]$$
$$[\forall s : \forall s' : [T_\phi^+(s, s') \wedge T_\phi^+(s', s) \Rightarrow SC_\phi(s, s')]]$$

Let $e \in \mathbb{B}^k$, we define $\alpha_e = \bigwedge_{e[i]=0} \neg a_i$. For each $e \in \mathbb{B}^k$, we define relations $T^+_{\phi\wedge\alpha_e}$ and $SC_{\phi\wedge\alpha_e}$ in the following clauses:

$$[\forall s : \forall s' : [T_\phi(s, s') \wedge R_{\alpha_e}(s) \wedge R_{\alpha_e}(s') \Rightarrow T_{\phi\wedge\alpha_e}(s, s')]]$$
$$[\forall s : \forall s' : [T_{\phi\wedge\alpha_e}(s, s') \Rightarrow T^+_{\phi\wedge\alpha_e}(s, s')]]$$

$$[\forall s : \forall s' : \forall s'' : [T^{+}_{\phi\wedge\alpha_e}(s,s') \wedge T^{+}_{\phi\wedge\alpha_e}(s',s'') \Rightarrow T^{+}_{\phi\wedge\alpha_e}(s,s'')]]$$
$$[\forall s : \forall s' : [T^{+}_{\phi\wedge\alpha_e}(s,s') \wedge T^{+}_{\phi\wedge\alpha_e}(s',s) \Rightarrow SC_{\phi\wedge\alpha_e}(s,s')]]$$

We define $Path_{sfair,\phi}$ in the following clauses:

$$[\forall s : [\bigvee_{e\in\mathbb{B}^k} [\bigwedge_{\substack{1\leq i\leq k\\ e(i)=1}} \exists s' : SC_{\phi\wedge\alpha_e}(s,s') \wedge R_{b_i}(s')] \Rightarrow SC_{sfair,\phi}(s)]$$
$$[\forall s : [\exists s' : T^{+}_{\phi}(s,s') \wedge SC_{sfair,\phi}(s')] \Rightarrow Path_{sfair,\phi}(s)]$$

Corollary 3.15 can be generalized to the case of $k > 1$ easily. In the next section, we briefly mention that fairness assumptions can be encoded in SFP as well. There, the exponential blow up does not occur.

3.2.3 Fairness in Succinct Fixed Point Logic

It is pointed out in [60] that almost all practical types of fairness constraints can be expressed using the canonical form $\bigwedge_{1\leq i\leq k}(\overset{\infty}{F}\, p_i \vee \overset{\infty}{G}\, q_i)$, where $\overset{\infty}{F}$ means "infinitely often" and $\overset{\infty}{G}$ means "almost always". It's shown in [6] that the canonical form can be characterized in μ-calculus formulas of alternation depth 2.

We reformulate their results using notations proposed in our setting. The canonical form mentioned above can be written as $fair = \bigwedge_{1\leq k\leq n}(\mathbf{GF}\phi_i \vee \mathbf{FG}\psi_i)$. We express unconditional, strong, and weak fairness constraints in the canonical form as follows:

Unconditional fairness $ufair = \bigwedge_{1\leq i\leq k} \mathbf{GF}\phi_i$ is already in canonical form.

Strong fairness constraints:

$$sfair = \bigwedge_{1\leq i\leq k} (\mathbf{GF}\phi_i \Rightarrow \mathbf{GF}\psi_i) \equiv \bigwedge_{1\leq i\leq k} (\mathbf{FG}\neg\phi_i \vee \mathbf{GF}\psi_i)$$

Weak fairness constraints:

$$wfair = \bigwedge_{1 \leq i \leq k} (\mathbf{FG}\phi_i \Rightarrow \mathbf{GF}\psi_i) \equiv \bigwedge_{1 \leq i \leq k} \mathbf{GF}(\neg\phi_i \vee \psi_i)$$

The canonical form can be encoded to the μ-calculus [6] by $\mathbf{E}\bigwedge_{1 \leq k \leq n}(\mathbf{GF}\phi_i \vee \mathbf{FG}\psi_i) = \mu Q_1.(\nu Q_2.\tau(Q_2) \vee \langle a \rangle Q_1)$ where $\tau(Q_2) = \bigwedge_{1 \leq i \leq k}(((\langle a \rangle \mu Q_3.\tau(Q_3)) \vee (\psi_i \wedge \langle a \rangle Q_2))$ and $\tau(Q_3) = ((\phi_i \wedge Q_2) \vee (Q_2 \wedge \langle a \rangle Q_3))$.

According to the results in Chapter 6, we can encode this μ-calculus formula in Succinct Fixed Point Logic.

3.3 Discussions

Tarjan's algorithm can be used to calculate strongly connected components in time complexity $\mathcal{O}(|S| + |T|)$ [76] for a finite Kripke structure $M = (S, T, L)$. Our specification for the set union of all nontrivial strongly connected sets involves calculating the transitive closure of transition relations, which yields a cubic time worse case complexity. It is worth considering how to derive a linear time worse case complexity specification for nontrivial strongly connected components.

It is shown in [77] that LTL model checking can be reduced to CTL model checking with fairness constraints. Actually, only unconditional fairness constraints are used there. Based on their reduction, an efficient symbolic LTL model checker has been developed. Our ALFP-based encoding works well for CTL model checking with unconditional fairness constraints. Hence, LTL model checking problem can also be properly expressed using ALFP constraints.

Chapter 4

Multi-valued Alternation-free Least Fixed Point Logic

Research work in [21] and the results in the previous chapter are two cases where ALFP has been used to analyze temporal properties of transition systems. In a more general point of view, ALFP can be used to perform two-valued static analysis for transition systems.

In this chapter, we show that it is possible to generalize this point of view from a 2-valued setting to a multi-valued setting. This means that the transition systems become multi-valued transition systems. Multi-valued transition systems can model uncertainty and inconsistency. Take modal transition systems (MTSs [52, 53]) as an example, where there are two transition relations; a *may* transition relation indicating the transitions that might be possible and a *must* transition relation indicating those transitions that must be possible. This is a form of multi-valued transition systems that has proved very useful for specifications of concurrent systems. We define the syntax and semantics of multi-valued ALFP based on a multi-valued structure which consists of a complete lattice and a total function defined over the complete lattice. We prove that the Moore family result carries over to this setting.

Rather than considering how to develop directly new solvers for multi-valued ALFP, we show that an analysis problem in multi-valued ALFP over a finite distributive multi-valued structure can be translated to a set of analysis problems in 2-valued ALFP. We give a time complexity result of Multi-valued ALFP based on our reduction method.

We point out that the 2-valued ALPF-based analysis for 2-valued CTL developed in Chapter 3 can be generalized to a multi-valued analysis by interpreting those analysis constraints in multi-valued ALFP. Many properties of 2-valued CTL are also preserved in our multi-valued analysis. Therefore, we also generalize the work in the previous chapter to the multi-valued setting. To show an application of this insight, we perform a three-valued ALFP-based analysis for the three-valued CTL model checking problem over Kripke modal transition systems.

The structure of this chapter is as follows. In Section 4.1, we first rephrase ALFP in two-valued setting and introduce the formal definition of two-valued transition systems. We show that two-valued transition systems can be encoded in ALFP naturally. The main consideration for this section is to set a scene, for example using new notations to define the semantics of ALFP, that can be later generalized to a multi-valued setting. Section 4.2 gives details of multi-valued ALFP and defines multi-valued transition systems. We reduce multi-valued ALFP into two-valued ALFP in Section 4.3. In Section 4.4, we interpret the ALFP-based analysis for two-valued CTL, developed in the previous chapter, using multi-valued ALFP. Section 4.5 is an application of multi-valued ALFP, where we focus on a three-valued setting. We introduce modal transition systems in Section 4.5.1. Section 4.5.2 introduces three-valued ALFP. Section 4.5.3 introduces three-valued CTL. We show in Section 4.5.4 that our three-valued ALFP-based analysis exactly characterizes the three-valued model checking for CTL over Kripke modal transition systems.

4.1 Two-valued Static Analysis

4.1.1 Two-valued ALFP

Two-valued ALFP is a special case of multi-valued ALFP. In the following, we briefly rephrase the syntax and semantics of two-valued ALFP using notations that is suited for the multi-valued setting.

The syntax of two-valued ALFP is given as follows. We use $pre \Rightarrow R(v_1, ..., v_n)$ instead of $pre \Rightarrow cl$ which is used in Section 2.2 to simplify our development, but this does not restrict the expressiveness (merely the succinctness) of ALFP.

$$\begin{array}{lll} v & ::= & c \mid x \\ pre & ::= & R(v_1, ..., v_n) \mid \neg R(v_1, ..., v_n) \mid pre_1 \wedge pre_2 \\ & & \mid pre_1 \vee pre_2 \mid \forall x: \ pre \mid \exists x: \ pre \\ cl & ::= & R(v_1, ..., v_n) \mid \mathbf{true} \mid cl_1 \wedge cl_2 \mid pre \Rightarrow R(v_1, ..., v_n) \mid \forall x: \ cl \end{array}$$

Let $Int^2 : \prod_k Rel_k \to \mathcal{U}^k \to \{true, false\}$ be a mapping where Rel_k is a finite alphabet of k-ary predicate symbols. The interpretation of ALFP is given in Table 4.1 in terms of satisfaction relations

$$(\varrho, \sigma) \ \underline{\mathtt{sat}}^2 \ pre \qquad \texttt{and} \qquad (\varrho, \sigma) \ \underline{\mathtt{sat}}^2 \ cl$$

where $\varrho \in Int^2$ maps each k-ary predicate symbol R to a 2-valued function and σ is an interpretation of variables.

Let us consider the mappings $S, S_1, S_2 : \mathcal{U}^k \to \{true, false\}$. We define that $S_1 \leq^2 S_2$ iff $\forall x \in \mathcal{U}^k : \neg S_1(x) \vee S_2(x)$. Given an index set I, the greatest lower bound is defined as $S = \bigwedge^2_{i \in I} S_i$ iff $\forall x \in \mathcal{U}^k : S(x) = \wedge^2_{i\in I} S_i(x)$. We write $<^2$ for the irreflexive part of $\leq^2$.

The lexicographic ordering $\leq^2_\sharp$ for the interpretations of relations can be defined as follows: $\varrho_1 \leq^2_\sharp \varrho_2$ if there exists a rank $i \in \{0, ..., r\}$ such that

$[(\varrho,\sigma)\ \underline{\mathtt{sat}}^2\ R(v_1,...,v_n)]$	$=$	$\varrho(R)(\sigma(v_1),...,\sigma(v_n))$
$[(\varrho,\sigma)\ \underline{\mathtt{sat}}^2\ \neg R(v_1,...,v_n)]$	$=$	$\neg\varrho(R)(\sigma(v_1),...,\sigma(v_n))$
$[(\varrho,\sigma)\ \underline{\mathtt{sat}}^2\ pre_1 \wedge pre_2]$	$=$	$[(\varrho,\sigma)\ \underline{\mathtt{sat}}^2\ pre_1]\ \wedge [(\varrho,\sigma)\ \underline{\mathtt{sat}}^2\ pre_2]$
$[(\varrho,\sigma)\ \underline{\mathtt{sat}}^2\ pre_1 \vee pre_2]$	$=$	$[(\varrho,\sigma)\ \underline{\mathtt{sat}}^2\ pre_1]\ \vee [(\varrho,\sigma)\ \underline{\mathtt{sat}}^2\ pre_2]$
$[(\varrho,\sigma)\ \underline{\mathtt{sat}}^2\ \forall x:\ pre]$	$=$	$\forall a \in \mathcal{U} : [(\varrho,\sigma[x\mapsto a])\ \underline{\mathtt{sat}}^2\ pre] = true$
$[(\varrho,\sigma)\ \underline{\mathtt{sat}}^2\ \exists x:\ pre]$	$=$	$\exists a \in \mathcal{U} : [(\varrho,\sigma[x\mapsto a])\ \underline{\mathtt{sat}}^2\ pre] = true$
$[(\varrho,\sigma)\ \underline{\mathtt{sat}}^2\ R(v_1,...,v_n)]$	$=$	$\varrho(R)(\sigma(v_1),...,\sigma(v_n))$
$[(\varrho,\sigma)\ \underline{\mathtt{sat}}^2\ \mathbf{true}]$	$=$	$\mathbf{true}$
$[(\varrho,\sigma)\ \underline{\mathtt{sat}}^2\ cl_1 \wedge cl_2]$	$=$	$[(\varrho,\sigma)\ \underline{\mathtt{sat}}^2\ cl_1]\ \wedge [(\varrho,\sigma)\ \underline{\mathtt{sat}}^2\ cl_2]$
$[(\varrho,\sigma)\ \underline{\mathtt{sat}}^2\ pre \Rightarrow R(v_1,...,v_n)]$	$=$	$\neg[(\varrho,\sigma)\ \underline{\mathtt{sat}}^2\ pre] \vee [(\varrho,\sigma)\ \underline{\mathtt{sat}}^2$ $R(v_1,...,v_n)]$
$[(\varrho,\sigma)\ \underline{\mathtt{sat}}^2\ \forall x:\ cl]$	$=$	$\forall a \in \mathcal{U} : [(\varrho,\sigma[x\mapsto a])\ \underline{\mathtt{sat}}^2\ cl] = true$

Table 4.1: Interpretation of Two-valued ALFP

- $\varrho_1(R) = \varrho_2(R)$ whenever $\mathtt{rank}_R < i$,
- $\varrho_1(R) \leq^2 \varrho_2(R)$ whenever $\mathtt{rank}_R = i$,
- either $i = r$ or $\varrho_1(R) <^2 \varrho_2(R)$ for some R with $\mathtt{rank}_R = i$.

We define $\varrho_1 \leq^2 \varrho_2$ to mean $\varrho_1(R) \leq^2 \varrho_2(R)$ for all $R \in \mathcal{R}$.

The set of interpretations of relations constitutes a complete lattice$(Int^2, \leq^2_\sharp)$. Proposition 2.6 is rephrased as follows:

Proposition 2.6 The set $\{\varrho | [(\varrho,\sigma_0)\ \underline{\mathtt{sat}}^2\ cl] = true\}$ is a Moore Family, i.e. is closed under greatest lower bounds, whenever *cl* is closed and stratified; the greatest lower bound $\wedge^2_\sharp\ \{\varrho | [(\varrho,\sigma_0)\ \underline{\mathtt{sat}}^2\ cl] = true\}$ is the *least* model of *cl*.

More generally, given ϱ_0 the set $\{\varrho | [(\varrho,\sigma_0)\ \underline{\mathtt{sat}}^2\ cl] = true \wedge \varrho_0 \leq^2 \varrho\}$ is a Moore Family and $\wedge^2_\sharp\ \{\varrho | [(\varrho,\sigma_0)\ \underline{\mathtt{sat}}^2\ cl] = true \wedge \varrho_0 \leq^2 \varrho\}$ is the least model.

4.1.2 Two-valued Transition Systems

A *transition system* (TS [10]) has the form $(S, S_0, \mathcal{A}, \rightarrow, P, V)$ where S is a set of states, $S_0 \subseteq S$ is a set of initial states, $\mathcal{A}$ is a set of actions, $\rightarrow \subseteq S \times \mathcal{A} \times S$

is a transition relation, P is a set of atomic propositions and $V : S \times P \to \{true, false\}$ is an interpretation that associates a truth value in $\{true, false\}$ with each atomic proposition in P for each state in S.

When $\mathcal{A}$ is non-empty and P is empty, a TS specializes to a *labeled transition system* (LTS) $(S, \mathcal{A}, \to)$. When $\mathcal{A}$ is a singleton and P is finite and non-empty, a TS specializes to a *Kripke structure* $(S, \to, P, V)$ except that we did not demand the transition relation to be total as is usually required.

To encode a TS $(S, S_0, \mathcal{A}, \to, P, V)$ into 2-valued ALFP, we assume that the universe $\mathcal{U} = S \cup \mathcal{A}$ and define corresponding predicates in ϱ_0 as follows:

- for each atomic proposition p over P, we define a predicate P_p such that $\varrho_0(P_p)(s) = V(s, p)$,
- for each subset Ω of $\mathcal{A}$, we define a relation Ω such that $\varrho_0(\Omega)(a) = true$ iff $a \in \Omega$,
- we define a ternary transition relation T such that $\varrho_0(T)(s, a, s') = true$ iff $(s, a, s') \in \to$, and
- we define a relation I such that $\varrho_0(I)(s) = true$ iff $s \in S_0$.

Example 4.1 *Let T and I, defined in ϱ_0, be the ternary transition relation and the set of initial states in a TS where S is finite, we can define a relation Reach to characterize the set of reachable states from S_0 by the following clause:*

$$(\forall s : I(s) \Rightarrow Reach(s)) \wedge (\forall s : \forall a : \forall s' : Reach(s) \wedge T(s, a, s') \Rightarrow Reach(s'))$$

The intention is that in the least solution ϱ to the above clause such that $\varrho_0 \leq^2 \varrho$, we know that $\varrho(Reach)(s) = true$ iff s is reachable from S_0. The above clause is obviously stratified by requiring $\textbf{rank}_T = 0, \textbf{rank}_I = 0$ and $\textbf{rank}_{Reach} = 0$.

4.2 Multi-valued Static Analysis

4.2.1 Multi-valued ALFP

The syntax of multi-valued ALFP is defined as follows. We still restrict ourselves to the *stratified* fragment of clauses. The notion of stratification remains the same in the multi-valued case.

$$
\begin{array}{lll}
v & ::= & c \mid x \\
pre & ::= & R(v_1, ..., v_n) \mid \neg R(v_1, ..., v_n) \mid pre_1 \wedge pre_2 \\
 & & \mid pre_1 \vee pre_2 \mid \forall x:\ pre \mid \exists x:\ pre \\
cl & ::= & \mathbf{true} \mid cl_1 \wedge cl_2 \mid pre \Rightarrow R(v_1, ..., v_n) \mid \forall x:\ cl
\end{array}
$$

In two-valued case, we have only two truth values, namely *true* and *false*, and $(\{true, false\}, \leq^2)$ constitutes a complete lattice. To generalize ALFP to a multi-valued setting, we introduce more than two truth values and require that these truth values constitute a complete lattice as well. Each complete lattice is equipped with two binary operators, namely least upper bound and greatest lower bound. These two operators can be used to interpret $\vee$ and $\wedge$ in the syntax of multi-valued ALFP, respectively. To interpret negation, we require that the complete lattice we considered is also equipped with a total function. We formalize this as *multi-valued structure* defined as follows.

Definition 4.1 A multi-valued structure is defined as $\mathcal{M} = (\mathcal{L}, \sim)$, where $\mathcal{L} = (L, \sqsubseteq) = (L, \sqsubseteq, \bigsqcup, \bigsqcap, \bot, \top)$ is a complete lattice and $\sim: L \to L$ is a total function.

The function $\sim$ intends to be interpreted over L as complement which maps each element in L to its unique complement. However, we point out that to prove the Moore family result of multi-valued ALFP, we only need to assume that $\sim$ is a total function in Definition 4.1 due to the notion of stratification.

Example 4.2 *Let $\mathcal{M} = (\mathcal{L}, \sim)$ be a multi-valued structure. Assume that $\mathcal{L} = (L, \sqsubseteq)$ is a De Morgan lattice [58]. Then, $\sim$ maps each element $l \in L$ to its unique complement $\sim l$ such that the following conditions hold: $\sim (l_1 \sqcup l_2) = \sim l_1 \sqcap \sim l_2$, $\sim (l_1 \sqcap l_2) = \sim l_1 \sqcup \sim l_2$, $\sim\sim l = l$ and $l_1 \sqsubseteq l_2$ iff $\sim l_1 \sqsupseteq l_2$.*

Let $\mathcal{M} = (\mathcal{L}, \sim)$ be a multi-valued structure and $Int : \prod_k Rel_k \to \mathcal{U}^k \to L$ be a mapping. We define the multi-valued interpretation of ALFP over $\mathcal{M}$ in Table 4.2 where $\varrho \in Int$ maps each k-ary predicate symbol R to a multi-valued function and σ is an interpretation of variables. Given σ_0 and a clause cl, a mapping ϱ satisfies cl if and only if $[(\varrho, \sigma_0)\ \underline{\mathtt{sat}}\ cl] = true$.

Notice that the truth value of $[(\varrho,\sigma)\ \underline{\texttt{sat}}\ pre]$ is multi-valued, but the truth value of $[(\varrho,\sigma)\ \underline{\texttt{sat}}\ cl]$ still remains two valued. Static analysis constraints are specified by cl. We are only interested in those interpretations which satisfy cl. As to whether ϱ satisfies cl or not, this is obviously a two valued problem. In the case of $pre \Rightarrow R(v_1,...,v_n)$, we think that ϱ correctly interprets $R(v_1,...,v_n)$ as long as $[(\varrho,\sigma)\ \underline{\texttt{sat}}\ pre] \sqsubseteq [(\varrho,\sigma)\ \underline{\texttt{sat}}\ R(v_1,...,v_n)]$ holds.

$[(\varrho,\sigma)\ \underline{\texttt{sat}}\ R(v_1,...,v_n)]$	$=$	$\varrho(R)(\sigma(v_1),...,\sigma(v_n))$
$[(\varrho,\sigma)\ \underline{\texttt{sat}}\ \neg R(v_1,...,v_n)]$	$=$	$\sim([[(\varrho,\sigma)\ \underline{\texttt{sat}}\ R(v_1,...,v_n)]])$
$[(\varrho,\sigma)\ \underline{\texttt{sat}}\ pre_1 \wedge pre_2]$	$=$	$[(\varrho,\sigma)\ \underline{\texttt{sat}}\ pre_1]\ \sqcap [(\varrho,\sigma)\ \underline{\texttt{sat}}\ pre_2]$
$[(\varrho,\sigma)\ \underline{\texttt{sat}}\ pre_1 \vee pre_2]$	$=$	$[(\varrho,\sigma)\ \underline{\texttt{sat}}\ pre_1]\ \sqcup [(\varrho,\sigma)\ \underline{\texttt{sat}}\ pre_2]$
$[(\varrho,\sigma)\ \underline{\texttt{sat}}\ \forall x:\ pre]$	$=$	$\bigsqcap_{a\in\mathcal{U}}\{[(\varrho,\sigma[x\mapsto a])\ \underline{\texttt{sat}}\ pre]\}$
$[(\varrho,\sigma)\ \underline{\texttt{sat}}\ \exists x:\ pre]$	$=$	$\bigsqcup_{a\in\mathcal{U}}\{[(\varrho,\sigma[x\mapsto a])\ \underline{\texttt{sat}}\ pre]\}$
$[(\varrho,\sigma)\ \underline{\texttt{sat}}\ \mathbf{true}]$	$=$	$\mathbf{true}$
$[(\varrho,\sigma)\ \underline{\texttt{sat}}\ cl_1 \wedge cl_2]$	$=$	$[(\varrho,\sigma)\ \underline{\texttt{sat}}\ cl_1]\ \wedge [(\varrho,\sigma)\ \underline{\texttt{sat}}\ cl_2]$
$[(\varrho,\sigma)\ \underline{\texttt{sat}}\ pre \Rightarrow R(v_1,...,v_n)]$	$=$	$\begin{cases} true & [(\varrho,\sigma)\ \underline{\texttt{sat}}\ pre] \sqsubseteq [(\varrho,\sigma)\ \underline{\texttt{sat}}\ R(v_1,...,v_n)] \\ false & \texttt{otherwise} \end{cases}$
$[(\varrho,\sigma)\ \underline{\texttt{sat}}\ \forall x:\ cl]$	$=$	$\forall a\in\mathcal{U}: [(\varrho,\sigma[x\mapsto a])\ \underline{\texttt{sat}}\ cl] = true$

Table 4.2: Multi-Valued Interpretation of ALFP

Let us consider the mappings $S, S_1, S_2 : \mathcal{U}^k \to L$. For the ordering $\sqsubseteq$, we have the following definitions. We define that $S_1 \sqsubseteq S_2$ iff $\forall x \in \mathcal{U}^k : S_1(x) \sqsubseteq S_2(x)$. Given an index set I, the greatest lower bound is defined as $S = \bigsqcap_{i\in I} S_i$ iff $\forall x \in \mathcal{U}^k : S(x) = \sqcap_{i\in I} S_i(x)$. We write $\sqsubset$ for the irreflexive part of $\sqsubseteq$.

The lexicographic ordering $\sqsubseteq_\sharp$ for the interpretations of relations is defined as follows: $\varrho_1 \sqsubseteq_\sharp \varrho_2$ if there exists a rank $i \in \{0,...,r\}$ for a stratified clause $cl = \bigwedge_{0\le i\le r} cl_i$ such that

- $\varrho_1(R) = \varrho_2(R)$ whenever $\texttt{rank}_R < i$,
- $\varrho_1(R) \sqsubseteq \varrho_2(R)$ whenever $\texttt{rank}_R = i$,
- either $i = r$ or $\varrho_1(R) \sqsubset \varrho_2(R)$ for some R with $\texttt{rank}_R = i$.

We also define $\varrho_1 \sqsubseteq \varrho_2$ to mean that $\varrho_1(R) \sqsubseteq \varrho_2(R)$ for all $R \in \mathcal{R}$.

The existence of the least model of multi-valued interpretations is guaranteed by the following theorem.

Theorem 4.2 *$\{\varrho|[(\varrho,\sigma_0)\ \underline{\texttt{sat}}\ cl]=true\}$ is a Moore Family with respect to $\sqsubseteq_\sharp$, i.e. is closed under greatest lower bounds, whenever cl is closed and stratified; the greatest lower bound $\sqcap_\sharp\ \{\varrho|[(\varrho,\sigma_0)\ \underline{\texttt{sat}}\ cl]=true\}$ is the least model of cl.*

More generally, given ϱ_0 the set $\{\varrho|[(\varrho,\sigma_0)\ \underline{\texttt{sat}}\ cl]=true\wedge\varrho_0\sqsubseteq\varrho\}$ is a Moore Family with respect to $\sqsubseteq_\sharp$ and $\sqcap_\sharp\ \{\varrho|[(\varrho,\sigma_0)\ \underline{\texttt{sat}}\ cl]=true\wedge\varrho_0\sqsubseteq\varrho\}$ is the least model.

Proof. In Appendix B. □

4.2.2 Multi-valued Transition Systems

A *multi-valued transition system* has the form $(S,S_0,\mathcal{A},\rightarrow,P,V)$ where S is a set of states, $S_0\subseteq S$ is a set of initial states, $\mathcal{A}$ is a set of actions, $\rightarrow: S\times\mathcal{A}\times S\rightarrow L$ is a multi-valued transition function, P is a set of atomic propositions and $V:S\times P\rightarrow L$ is a multi-valued interpretation that associates a value in L with each atomic proposition in P for each state in S.

To encode a multi-valued TS $(S,S_0,\mathcal{A},\rightarrow,P,V)$ into multi-valued ALFP, we assume that the universe $\mathcal{U}=S\cup\mathcal{A}$ and define corresponding predicates in ϱ_0 as follows:

- for each atomic proposition p over P, we define a predicate P_p such that $\varrho_0(P_p)(s)=V(s,p)$,
- for each subset Ω of $\mathcal{A}$, we define a relation Ω such that $\varrho_0(\Omega)(a)=\top$ iff $a\in\Omega$,
- we define a ternary transition relation T such that $\varrho_0(T)(s,a,s')=\rightarrow(s,a,s')$, and
- we define a relation I such that $\varrho_0(I)(s)=\top$ iff $s\in S_0$.

Example 4.3 *Let $M=(S,S_0,\mathcal{A},\rightarrow,P,V)$ be a multi-valued transition system where S is finite , $\mathcal{A}=\{a\}$ is a singleton and $\rightarrow(s,a,s')$ denote the reliability* *of the connection between s and s'. Let T, defined in ϱ_0, be the binary transition relation of M such that $\varrho_0(T)(s,s') = \rightarrow (s,a,s')$. Assume that s_t is a target state in M. We specify a relation Reliability by the following clause and define that $\varrho_0(Reliability)(s) = \top$ iff $s = s_t$:*

$$\forall s : [\exists s' : T(s,s') \wedge Reliability(s')] \Rightarrow Reliability(s)$$

For the least solution ϱ to the above clause such that $\varrho_0 \sqsubseteq \varrho$, we know that $\varrho(Reliability)(s)$ can approximate the reliability of going from s to s_t. The above clause is also obviously stratified by requiring $\mathbf{rank}_T = 0$ and $\mathbf{rank}_{Reliability} = 0$.

4.3 Reducing Multi-valued ALFP to Two-valued ALFP

In this section, we show that multi-valued ALFP over a finite distributive multi-valued structure can be reduced to two-valued ALFP. Recall that a lattice is distributive iff $l_1 \sqcup (l_2 \sqcap l_3) = (l_1 \sqcup l_2) \sqcap (l_1 \sqcup l_3)$ and $l_1 \sqcap (l_2 \sqcup l_3) = (l_1 \sqcap l_2) \sqcup (l_1 \sqcap l_3)$. We define a finite distributive multi-valued structure as follows:

Definition 4.3 A finite distributive multi-valued structure is a multi-valued structure $\mathcal{M} = (\mathcal{L}, \sim)$, where $\mathcal{L} = (L, \sqsubseteq)$ is a finite distributive lattice.

Let us first explain the link between ALFP and negation-free ALFP.

Assume that $cl = \bigwedge_{0 \leq i \leq r} cl_i$ is a stratified clause. From the notion of stratification, we know that in the clause cl_i all negatively used relations are defined either in an initial model ϱ_0 or by $\bigwedge_{0 \leq j \leq i-1} cl_j$. We use $\varrho(i)$ to denote the interpretation of relations of rank i $(0 \leq i \leq n)$ in ϱ, and we define ϱ_i by $\varrho_i = \varrho(0) \cup \varrho(1) \cup ... \cup \varrho(i)$. Let ϱ_{i-1} be the least model to $\bigwedge_{0 \leq j \leq i-1} cl_j$ subjected to $\varrho_0 \sqsubseteq \varrho_{i-1}$ and ϱ_{i-1}^{neg} be a new interpretation. For each relation R defined in ϱ_{i-1}, we define a new relation $R^{\neg}$ in ϱ_{i-1}^{neg} by $\varrho_{i-1}^{neg}(R^{\neg}) = \sim \varrho_{i-1}(R)$.

We translate cl_i to a negation-free clause cl_i^+ by substituting each negative use of a relation R in cl_i, i.e of the form $\neg R(v_1, ..., v_n)$, with $R^{\neg}(v_1, ..., v_n)$. Let $\varrho_i = \sqcap\{\varrho_i | [(\varrho_i, \sigma)\ \underline{\mathtt{sat}}\ cl_i] = true \wedge \varrho_{i-1} \sqsubseteq \varrho_i\}$ and $\varrho_i' = \sqcap\{\varrho_i' | [(\varrho_{i-1}^{neg} \cup \varrho_i', \sigma)\ \underline{\mathtt{sat}}\ cl_i^+] =$

$true \wedge \varrho_{i-1} \sqsubseteq \varrho'_i\}$. It is easy to see that $\varrho_i = \varrho'_i$. Therefore, the problem of calculating the least model of $cl = \bigwedge_{0\leq i\leq r} cl_i$ reduces to evaluating corresponding negation-free ALFP clauses $cl = \bigwedge_{0\leq i\leq r} cl_i^+$ $(0 \leq i \leq r)$.

From above, we know that reducing multi-valued ALFP to 2-valued ALFP boils down to reducing negation-free multi-valued ALFP to 2-valued ALFP. We now consider the negation-free fragment of multi-valued ALFP clauses. When cl is negation-free, we can use $\wedge^2$ (resp. $\sqcap$) instead of $\wedge^2_\sharp$ (resp. $\sqcap_\sharp$) to denote the same meaning.

First, we intend to establish the link between a multi-valued interpretation ϱ and a set of two-valued interpretations $(\varrho^{x_1}, ..., \varrho^{x_n})$, where $x_i \in L$ $(1 \leq i \leq n)$. The intention is that $\forall s \in \mathcal{U}^k, \forall R \in \mathcal{R}, \forall 1 \leq i \leq n : x_i \sqsubseteq \varrho(R)(s)$ iff $\varrho^{x_i}(R)(s) = true$ and that $\varrho(R)(s) = \bigsqcup\{x_i | \varrho^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\}$. To this end, we choose join-irreducible elements of L as $x_1, ..., x_n$. The definition of join-irreducible elements [59] is given as follows.

Definition 4.4 Let $\mathcal{L} = (L, \sqsubseteq)$ be a lattice. An element $x \in L$ is a join-irreducible element if x is not bottom (in case L has a bottom) and $x = y \sqcup z$ implies $x = y$ or $x = z$ for all $y, z \in L$. We use $\mathcal{J}(\mathcal{L})$ to denote the set of join-irreducible elements of L.

We introduce the following definition to impose a constraint on $(\varrho^{x_1}, ..., \varrho^{x_n})$. The idea is that we need to make sure those interpretations do not contain contradictory information.

Definition 4.5 A tuple $(\varrho^{x_1}, ..., \varrho^{x_n})$ where $\varrho^{x_i} \in Int^2 (1 \leq i \leq n)$ is called consistent, denoted by $\mathcal{C}((\varrho^{x_1}, ..., \varrho^{x_n}))$, iff $x_i \sqsupseteq x_j$ implies $\varrho^{x_i} \sqsubseteq \varrho^{x_j}$ $(1 \leq i, j \leq n)$.

Given a finite distributive multi-valued structure $\mathcal{M} = (\mathcal{L}, \sim)$ and $\mathcal{J}(\mathcal{L}) = \{x_1, ..., x_n\}$. We now construct two isomorphic posets $(\mathcal{I}, \sqsubseteq)$ and $(\mathcal{I}^2, \leq^2)$, where $\mathcal{I} = Int$, $\mathcal{I}^2 = \{(\varrho^{x_1}, ..., \varrho^{x_n}) | \forall 1 \leq i \leq n : \varrho^{x_i} \in Int^2 \wedge \mathcal{C}((\varrho^{x_1}, ..., \varrho^{x_n}))\}$ and $\leq^2$ also denotes its pointwise extension. The isomorphism is guaranteed by Lemma 4.7 and Corollary 4.8 below.

Definition 4.6 We define the function $\mathbf{f} : \mathcal{I} \to \mathcal{I}^2$ by $\mathbf{f}(\varrho) = (\varrho^{x_1}, ..., \varrho^{x_n})$ where $\forall s \in \mathcal{U}^k, \forall R \in \mathcal{R}, \forall 1 \leq i \leq n : \varrho^{x_i}(R)(s) = true$ iff $x_i \sqsubseteq \varrho(R)(s)$. We define the function $\mathbf{b} : \mathcal{I}^2 \to \mathcal{I}$ by $\mathbf{b}((\varrho^{x_1}, ..., \varrho^{x_n})) = \varrho$ where $\forall s \in \mathcal{U}^k, \forall R \in \mathcal{R}$: $\varrho(R)(s) = \bigsqcup\{x_i | \varrho^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\}$.

Lemma 4.7 *The functions* $\mathbf{f}$ *and* $\mathbf{b}$ *are monotone,* $\mathbf{b} \circ \mathbf{f} = id_{\mathcal{I}}$ *and* $\mathbf{f} \circ \mathbf{b} = id_{\mathcal{I}^2}$ *where* $id_{\mathcal{I}}$ *and* $id_{\mathcal{I}^2}$ *are the identity functions over* $\mathcal{I}$ *and* $\mathcal{I}^2$ *respectively.*

Proof. In Appendix B. □

Corollary 4.8 *The posets* $(\mathcal{I}, \sqsubseteq)$ *and* $(\mathcal{I}^2, \leq^2)$ *are isomorphic.*

Proof. It follows directly from Lemma 4.7 and Definition 2.4 given in Section 2.1. □

Second, we consider the link between a multi-valued model ϱ of a negation-free multi-valued ALFP clause cl and a tuple of two-valued models $(\varrho^{x_1}, ..., \varrho^{x_n})$ of cl. Given ϱ_0 and σ_0. Let $\mathcal{I}_{cl,\varrho_0,\sigma_0} = \{\varrho | [(\varrho, \sigma_0)\ \underline{\mathtt{sat}}\ cl] = true \wedge \varrho_0 \sqsubseteq \varrho\}$ and $\mathcal{I}^2_{cl,\varrho_0,\sigma_0} = \{(\varrho^{x_1}, ..., \varrho^{x_n}) | \forall 1 \leq i \leq n : [(\varrho^{x_i}, \sigma_0)\ \underline{\mathtt{sat}}^2\ cl] = true \wedge \mathbf{f}(\varrho_0) \leq^2 (\varrho^{x_1}, ..., \varrho^{x_n}) \wedge \mathcal{C}((\varrho^{x_1}, ..., \varrho^{x_n}))\}$. We define $\mathbf{f}(\mathcal{I}_{cl,\varrho_0,\sigma_0}) = \{\mathbf{f}(\varrho) | \varrho \in \mathcal{I}_{cl,\varrho_0,\sigma_0}\}$ and $\mathbf{b}(\mathcal{I}^2_{cl,\varrho_0,\sigma_0}) = \{\mathbf{b}((\varrho^{x_1}, ..., \varrho^{x_n})) | (\varrho^{x_1}, ..., \varrho^{x_n}) \in \mathcal{I}^2_{cl,\varrho_0,\sigma_0}\}$. Now we focus on the posets $(\mathcal{I}_{cl,\varrho_0,\sigma_0}, \sqsubseteq)$ and $(\mathcal{I}^2_{cl,\varrho_0,\sigma_0}, \leq^2)$. It's obvious that $\mathcal{I}_{cl,\varrho_0,\sigma_0} \subseteq \mathcal{I}$ and $\mathcal{I}^2_{cl,\varrho_0,\sigma_0} \subseteq \mathcal{I}^2$. We can also prove that $\mathbf{f}(\mathcal{I}_{cl,\varrho_0,\sigma_0}) \subseteq \mathcal{I}^2_{cl,\varrho_0,\sigma_0}$ and $\mathbf{b}(\mathcal{I}^2_{cl,\varrho_0,\sigma_0}) \subseteq \mathcal{I}_{cl,\varrho_0,\sigma_0}$. Then, from Corollary 4.8, we have the following theorem.

Theorem 4.9 *Given* ϱ_0 *and a negation-free multi-valued ALFP clause* cl*. The two posets* $(\mathcal{I}_{cl,\varrho_0,\sigma_0}, \sqsubseteq)$ *and* $(\mathcal{I}^2_{cl,\varrho_0,\sigma_0}, \leq^2)$ *are isomorphic.*

Proof. In Appendix B. □

The following lemma tells that if $(\varrho^{x_1}, ..., \varrho^{x_n}) = \wedge^2\{(\varrho^{x_1}, ..., \varrho^{x_n}) | \forall 1 \leq i \leq n : [(\varrho^{x_i}, \sigma_0)\ \underline{\mathtt{sat}}^2\ cl] = true \wedge \mathbf{f}(\varrho_0) \leq^2 (\varrho^{x_1}, ..., \varrho^{x_n})\}$, we then know that $\mathcal{C}((\varrho^{x_1}, ..., \varrho^{x_n}))$ holds.

Lemma 4.10 *Let* $\mathcal{M} = (\mathcal{L}, \sim)$ *be a finite distributive multi-valued structure. Then* $\wedge^2\mathcal{I}^2_{cl,\varrho_0,\sigma_0} = \wedge^2\{(\varrho^{x_1}, ..., \varrho^{x_n}) | \forall 1 \leq i \leq n : [(\varrho^{x_i}, \sigma_0)\ \underline{\mathtt{sat}}^2\ cl] = true \wedge \mathbf{f}(\varrho_0) \leq^2 (\varrho^{x_1}, ..., \varrho^{x_n})\}$.

PROOF. In Appendix B. □

From Theorem 4.9 and Lemma 4.10, we have the following theorem which is the main theorem of this section.

THEOREM 4.11 *Let $\mathcal{M} = (\mathcal{L}, \sim)$ be a finite distributive multi-valued structure, $\mathcal{J}(\mathcal{L}) = \{x_1, ..., x_n\}$, $\varrho_0 \in \mathcal{I}$ and cl be a negation-free multi-valued ALFP clause. Let $\varrho = \sqcap \mathcal{I}_{cl,\varrho_0,\sigma_0}$, $\varrho^{x_i} = \wedge^2\{\varrho^{x_i} | [(\varrho^{x_i}, \sigma_0)\ \underline{\mathbf{sat}}^2\ cl] = true \wedge \varrho_0^{x_i} \leq^2 \varrho^{x_i}\}$ where $1 \leq i \leq n$ and $\mathbf{f}(\varrho_0) = (\varrho_0^{x_1}, ..., \varrho_0^{x_n})$. We then have $\varrho = \mathbf{b}((\varrho^{x_1}, ..., \varrho^{x_n}))$.*

PROOF. It's obvious from Theorem 4.9 and Lemma 4.10. □

Complexity of Multi-valued ALFP: According to [29], the least solution ϱ of a two-valued ALFP clause cl such that $\varrho_0 \leq^2 \varrho$, where ϱ_0 is an initial interpretation, can be computed in time $\mathcal{O}(\sharp\varrho + N^r \cdot n)$ where N is the size of the universe, n is the size of cl, r is the maximal nesting depth of quantifiers in cl and $\sharp\varrho$ is the sum of cardinalities of predicates $\varrho(R)$.

A multi-valued ALFP clause cl is also a two-valued ALFP clause. Assume that we evaluate cl over a finite distributive multi-valued structure $\mathcal{M} = (\mathcal{L}, \sim)$, where $\mathcal{L} = (L, \sqsubseteq)$ and $\mathcal{J}(\mathcal{L})$ is the join-irreducible elements of L. The least solution ϱ of cl such that $\varrho_0 \sqsubseteq \varrho$, where ϱ_0 is an initial interpretation, can be computed in time $\mathcal{O}((\sharp\varrho + N^r \cdot n) \cdot |\mathcal{J}(\mathcal{L})|)$, where $|\mathcal{J}(\mathcal{L})|$ is the number of elements in $\mathcal{J}(\mathcal{L})$. This means multi-valued ALFP can be evaluated in time linear to $|\mathcal{J}(\mathcal{L})|$. We only need to run the 2-valued succinct solver $|\mathcal{J}(\mathcal{L})|$ times. The worst running time seems occurs when $\mathcal{L}$ is a linear order. In that case, we can check the elements in the middle of the lattice and then recursively check the upper or lower half according to the analysis result by using binary search. In this way, we only need to run the succinct solver $\mathcal{O}(log(|\mathcal{J}(\mathcal{L})|))$ times.

4.4 Static Analysis of Multi-valued Transition Systems

The analysis developed in Table 3.1 naturally generalizes to a multi-valued analysis of CTL over a multi-valued TS when using multi-valued ALFP to interpret those ALFP clauses. Notice that we need to modify the analysis for the case of the CTL formula **true**. This is because to make sure that clauses are two valued in multi-valued ALFP, assertions of relations are not allowed in the syntax. However, we can always assert a relation in an initial interpretation ϱ_0. Therefore, this does not limit the expressiveness of multi-valued ALFP.

We list our multi-valued analysis for CTL formulas in Table 4.3. In the case of $\vec{R} \vdash \mathbf{true}$, $True$ is a predefined relation in ϱ_0 such that $\varrho_0(True)(s) = \top$ for all states s.

$$
\begin{array}{lll}
\vec{R} \vdash \mathbf{true} & \underline{\texttt{iff}} & [\forall s : True(s) \Rightarrow R_{true}(s)] \\
\vec{R} \vdash p & \underline{\texttt{iff}} & [\forall s : P_p(s) \Rightarrow R_p(s)] \\
\vec{R} \vdash \phi_1 \vee \phi_2 & \underline{\texttt{iff}} & \vec{R} \vdash \phi_1 \wedge \vec{R} \vdash \phi_2 \wedge \\
 & & [\forall s : R_{\phi_1}(s) \vee R_{\phi_2}(s) \Rightarrow R_{\phi_1 \vee \phi_2}(s)] \\
\vec{R} \vdash \neg\phi & \underline{\texttt{iff}} & \vec{R} \vdash \phi \wedge \\
 & & [\forall s : \neg R_\phi(s) \Rightarrow R_{\neg\phi}(s)] \\
\vec{R} \vdash \mathbf{EX}\phi & \underline{\texttt{iff}} & \vec{R} \vdash \phi \wedge \\
 & & [\forall s : [\exists s' : T(s,s') \wedge R_\phi(s')] \Rightarrow R_{\mathbf{EX}\phi}(s)] \\
\vec{R} \vdash \mathbf{E}[\phi_1 \mathbf{U} \phi_2] & \underline{\texttt{iff}} & \vec{R} \vdash \phi_1 \wedge \vec{R} \vdash \phi_2 \wedge \\
 & & [\forall s : R_{\phi_2}(s) \Rightarrow R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}(s)] \wedge \\
 & & [\forall s : [\exists s' : T(s,s') \wedge R_{\phi_1}(s) \wedge R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}(s')] \\
 & & \Rightarrow R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}(s)] \\
\vec{R} \vdash \mathbf{AF}\phi & \underline{\texttt{iff}} & \vec{R} \vdash \phi \wedge \\
 & & [\forall s : R_\phi(s) \Rightarrow R_{\mathbf{AF}\phi}(s)] \wedge \\
 & & [\forall s : [\forall s' : \neg T(s,s') \vee R_{\mathbf{AF}\phi}(s')] \Rightarrow R_{\mathbf{AF}\phi}(s)]
\end{array}
$$

Table 4.3: CTL in Multi-valued ALFP

Let $N = (S, S_0, \rightarrow, P, V)$ be multi-valued TS, where S is finite and P is finite and non-empty. We assume that N is "total" by requiring that $\forall s \in S, \exists s' :\rightarrow (s,s') \neq \bot$. As is introduced in Section 4.2.2, N can be encoded into multi-valued ALFP and we define corresponding predicates in ϱ_0.

We choose to evaluate multi-valued ALFP over a finite distributive multi-valued structures $\mathcal{M} = (\mathcal{L}, \sim)$, where the negation $\sim$ is defined the same as in Example 4.2 such that $\sim$ is anti-monotonic, preserves De Morgan laws and $\sim\sim l = l$ $(l \in L)$. The purpose of restricting ourselves to such a multi-valued structure is that: (1) we can reduce our multi-valued analysis to two-valued analysis using the method explained in Section 4.3; (2) many properties in two-valued CTL, i.e. equivalences and dualities of CTL formulas, can be preserved in our multi-valued analysis.

For a multi-valued interpretation ϱ, $\varrho(R_\phi)$ maps a state s to a lattice element in L. In the following, we assume that ϱ is the least solution to $\vec{R} \vdash \phi$ subject to $\varrho_0 \sqsubseteq \varrho$ and explain Table 4.3 in multi-valued setting briefly.

In the case of **true**, it's obvious that $\varrho(R_{true})$ maps each state s to $\top$ according to the semantics of multi-valued ALFP. For the atomic proposition p, we know from the constraint $\forall s : P_p(s) \Rightarrow R_p(s)$ that $\varrho(P_p) = \varrho(R_p)$. The clauses for boolean operators $\vee, \wedge$ and $\neg$ follow the same pattern so we just explain one of them, namely disjunction $\phi_1 \vee \phi_2$. The judgements $\vec{R} \vdash \phi_1$ and $\vec{R} \vdash \phi_2$ ensure that for a relation $R_{\phi'}$ corresponding to a subformula ϕ' of ϕ_1 or ϕ_2, $\varrho(R_{\phi'})$ maps states to corresponding elements in L as intended. The clause $\forall s : R_{\phi_1}(s) \vee R_{\phi_2}(s) \Rightarrow R_{\phi_1 \vee \phi_2}(s)$ ensures that $\varrho(R_{\phi_1}) \sqcup \varrho(R_{\phi_2}) = \varrho(R_{\phi_1 \vee \phi_2})$. We can easily prove that De Morgan's law for boolean formulas are preserved in our multi-value analysis.

In the case of $\mathbf{EX}\phi$, the first conjunct ensures that for a relation $R_{\phi'}$ corresponding to a subformula ϕ' of ϕ, $\varrho(R_{\phi'})$ carries the intended analysis result for ϕ'. The second conjunct ensures that $\sqcup_{s' \in S}(\varrho(T)(s, s') \sqcap \varrho(R_\phi)(s')) = \varrho(R_{\mathbf{EX}\phi})(s)$ hold for any state s. As can been seen in the following, this helps to preserve properties in two-valued CTL in our analysis.

In the case of $\mathbf{E}[\phi_1 \mathbf{U} \phi_2]$, the judgements $\vec{R} \vdash \phi_1$ and $\vec{R} \vdash \phi_2$ play the same role as in the case of $\phi_1 \vee \phi_2$. From the other two conjuncts and the definition of $\vec{R} \vdash \mathbf{EX}\phi$, we know that for any state s we have $\varrho(R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]})(s) = \varrho(R_{\phi_2})(s) \sqcup (\bigsqcup_{s' \in S}(\varrho(T)(s, s') \sqcap \varrho(R_{\phi_1})(s) \sqcap \varrho(R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]})(s'))) = \varrho(R_{\phi_2})(s) \sqcup (\bigsqcup_{s' \in S} \varrho(R_{\phi_1})(s)) \sqcap \bigsqcup_{s' \in S}(\varrho(T)(s, s') \sqcap \varrho(R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]})(s')) = \varrho(R_{\phi_2})(s) \sqcup (\varrho(R_{\phi_1})(s) \sqcap \varrho(R_{\mathbf{EXE}[\phi_1 \mathbf{U} \phi_2]})(s))$. This means the equivalence $\mathbf{E}[\phi_1 \mathbf{U} \phi_2] \equiv \phi_2 \vee (\phi_1 \wedge \mathbf{EXE}[\phi_1 \mathbf{U} \phi_2])$ in two-valued CTL is preserved in our analysis.

To explain the case of $\mathbf{AF}\phi$, we first give the definition of $\vec{R} \vdash \mathbf{AX}\phi$ as follows, which ensures that $\varrho(R_{\mathbf{AX}\phi}) = \sim \varrho(R_{\mathbf{EX}\neg\phi})$.

$$\begin{array}{lll} \vec{R} \vdash \mathbf{AX}\phi & \underline{\texttt{iff}} & \vec{R} \vdash \mathbf{EX}\neg\phi \wedge \\ & & [\forall s : \neg R_{\mathbf{EX}\neg\phi}(s) \Rightarrow R_{\mathbf{AX}\phi}(s)] \end{array}$$

In the case of $\mathbf{AF}\phi$, the first conjunct plays the same role as in the case of

$\mathbf{EX}\phi$. From the other two conjuncts, the definition of $\vec{R} \vdash \neg\phi$ and the definition of $\vec{R} \vdash \mathbf{AX}\phi$, we see that for any state s we have $\varrho(R_{\mathbf{AF}\phi})(s) = \varrho(R_\phi)(s) \sqcup (\bigsqcap_{s' \in S}(\sim \varrho(T)(s,s') \sqcup \varrho(R_{\mathbf{AF}\phi})(s'))) = \varrho(R_\phi)(s) \sqcup (\sim \bigsqcup_{s' \in S}(\varrho(T)(s,s') \sqcap \sim \varrho(R_{\mathbf{AF}\phi})(s'))) = \varrho(R_\phi)(s) \sqcup (\sim \bigsqcup_{s' \in S}(\varrho(T)(s,s') \sqcap \varrho(R_{\neg\mathbf{AF}\phi})(s'))) = \varrho(R_\phi)(s) \sqcup \sim \varrho(R_{\mathbf{EX}\neg\mathbf{AF}\phi})(s) = \varrho(R_\phi)(s) \sqcup \varrho(R_{\mathbf{AXAF}\phi})(s)$. This means the equivalence $\mathbf{AF}\phi \equiv \phi \vee \mathbf{AXAF}\phi$ in two-valued CTL is preserved in our analysis.

We define our analysis for the case of $\mathbf{EF}\phi, \mathbf{A}[\phi_1 \mathbf{U} \phi_2], \mathbf{EG}\phi$ and $\mathbf{AG}\phi$ as follows. One can verify that the equivalences introduced in Section 2.3.2 is also preserved in our analysis.

$$
\begin{array}{lll}
\vec{R} \vdash \mathbf{EF}\phi & \underline{\texttt{iff}} & \vec{R} \vdash \mathbf{E}[\mathbf{true}\mathbf{U}\phi] \\
\vec{R} \vdash \mathbf{A}[\phi_1\mathbf{U}\phi_2] & \underline{\texttt{iff}} & \vec{R} \vdash \mathbf{E}[\neg\phi_2\mathbf{U}(\neg\phi_1 \wedge \neg\phi_2)] \wedge \vec{R} \vdash \mathbf{AF}\phi_2 \wedge \\
 & & [\forall s : \neg R_{\mathbf{E}[\neg\phi_2\mathbf{U}(\neg\phi_1\wedge\neg\phi_2)]}(s) \wedge R_{\mathbf{AF}\phi_2}(s) \Rightarrow R_{\mathbf{A}[\phi_1\mathbf{U}\phi_2]}(s)] \\
\vec{R} \vdash \mathbf{EG}\phi & \underline{\texttt{iff}} & \vec{R} \vdash \mathbf{AF}\neg\phi \wedge \\
 & & [\forall s : \neg R_{\mathbf{AF}\neg\phi}(s) \Rightarrow R_{\mathbf{EG}\phi}(s)] \\
\vec{R} \vdash \mathbf{AG}\phi & \underline{\texttt{iff}} & \vec{R} \vdash \mathbf{E}[\mathbf{true}\mathbf{U}\neg\phi] \wedge \\
 & & [\forall s : \neg R_{\mathbf{E}[\mathbf{true}\mathbf{U}\neg\phi]}(s) \Rightarrow R_{\mathbf{AG}\phi}(s)]
\end{array}
$$

Remark: In this section, we have generalized our work on the analysis of 2-valued CTL to a multi-valued setting by evaluating those clauses using the semantics of multi-valued ALFP. We have shown that many properties of 2-valued CTL can be preserved in our multi-valued analysis so that our multi-valued analysis for CTL is a satisfactory analysis approach. A multi-valued semantics for CTL has been proposed in [43]. It would be an interesting work to compare our multi-valued analysis result with their semantics of multi-valued CTL.

On the other hand, we also want to point out that we do not intend to, in general, obtain a multi-valued analysis by interpreting 2-valued ALFP clauses using multi-valued semantics of ALFP. However, this could be a first try.

4.5 Application to Modal Transition Systems

This section is an application of multi-valued ALFP. We still focus on analyzing temporal properties of transition systems. We show that the three-valued CTL model checking problem over Kripke modal transition systems can be encoded into three-valued ALFP. This also concretizes the insight proposed in the previous section that our static analysis developed for two-valued CTL can be lifted to multi-valued settings.

4.5.1 Modal Transition Systems

Three-valued modeling formalisms are useful techniques in reasoning about system properties. *Partial Kripke structures* [49] support the modeling of incomplete state space of a system. *Modal transition systems* (MTSs [52, 53]) provide specifications of necessary behaviors and possible behaviors, which explicitly characterizes uncertainties of systems, and allow for the validation as well as refutation of system properties. *Kripke modal transition systems* (Kripke MTSs) [40, 41, 56] is a generalization of MTSs.

Research in [78] has compared the above three types of three-valued modeling formalisms and shown that they have the same expressiveness. We give the definition of Kripke MTSs as follows.

Definition 4.12 (Kripke Modal Transition System) A Kripke Modal Transition System (KMTS) over a finite atomic propositions set $\boldsymbol{P}$ is a tuple $M = (S, S_0, \xrightarrow{must}, \xrightarrow{may}, L)$, where S is a nonempty finite set of states, $S_0 \subseteq S$ is a set of initial states, $\xrightarrow{may} \subseteq S \times S$ and $\xrightarrow{must} \subseteq S \times S$ are transition relations such that the relation $\xrightarrow{may}$ is total and $\xrightarrow{must} \subseteq \xrightarrow{may}$, and $L : S \times P \to \{true, \bot, false\}$ is an interpretation that associates a truth value in $\{true, \bot, false\}$ with each atomic proposition in P for each state in S.

Transitions in $\xrightarrow{must}$ and $\xrightarrow{may}$ are *must* transitions and *may* transitions respectively. We write $s \xrightarrow{must} s'$ (resp. $s \xrightarrow{may} s'$) when $(s, s') \in \xrightarrow{must}$ (resp. $(s, s') \in \xrightarrow{may}$). A *must* (resp. *may*) path from state s is a maximal sequence of states $\pi = s_0, s_1 ...$ such that $s = s_0$ and for each pair of consecutive states s_i, s_{i+1} in π, we have $s_i \xrightarrow{must} s_{i+1}$ (resp. $s_i \xrightarrow{may} s_{i+1}$). Since $\xrightarrow{may}$ is total, every may path is infinite. A must path can be finite since $\xrightarrow{must}$ is not necessarily total. By maximality we mean that it's not possible to extend the path by any other transition of the same type. We use $|\pi|$ to denote the length of the path π. If π is an infinite path, then $|\pi| = \infty$. If $\pi = s_0, s_1 ... s_n$, then $|\pi| = n+1$. For a finite path $\pi = s_0, s_1 ... s_n$, we use $\pi[k] (0 \leq k \leq n)$ to denote the $(k+1)$th state s_k of π. For an infinite path $\pi = s_0, s_1 ...$, we use $\pi[k](0 \leq k)$ to denote the $(k+1)$th state s_k of π as well. We say that s' is a must (resp. may) successor of s if $s \xrightarrow{must} s'$ (resp. $s \xrightarrow{may} s'$).

Reasoning about Kripke MTSs requires 3-valued logical formalisms. The work in [41] defines a game-based three-valued CTL model checking over Kripke MTSs. In the next section, we introduce 3-valued ALFP as an application of multi-valued ALFP. In 3-valued setting, we can characterize uncertainties of system behaviors as *unknown* information. The application of the 3-valued ALFP to the analysis of Kripke MTSs is introduced in Section 4.5.4.

4.5.2 Three-valued ALFP

In this section, we define three-valued ALFP. The idea here is that we reuse the syntax of multi-valued ALFP defined in Section 4.2.1 and define three-valued semantics based on Kleene's three-valued proposition logic [48].

The syntax of 3-valued ALFP is defined in the following. As in multi-valued ALFP, we also restrict ourselves to its stratified fragment.

$$
\begin{array}{lcl}
v & ::= & c \mid x \\
pre & ::= & R(v_1, ..., v_n) \mid \neg R(v_1, ..., v_n) \mid pre_1 \wedge pre_2 \\
 & & \mid pre_1 \vee pre_2 \mid \forall x : \ pre \mid \exists x : \ pre \\
cl & ::= & \mathbf{true} \mid cl_1 \wedge cl_2 \mid pre \Rightarrow R(v_1, ..., v_n) \mid \forall x : \ cl
\end{array}
$$

It remains to define the semantics of 3-valued ALFP. We briefly recall Kleene's 3-valued propositional logic in the following.

In Kleene's 3-valued propositional logic [48], $\perp$ is understood as "unknown". Conjunction $\wedge^3$ and disjunction $\vee^3$ are defined as the minimum and maximum of its arguments according to the *truth ordering*, $false \leq^3 \perp \leq^3 true$ (see Figure 4.1). Also we have $\bigwedge^3_{i \in \emptyset} f_i = true$ and $\bigvee^3_{i \in \emptyset} f_i = false$. Negation $\neg^3$ maps $true$ to $false$, $false$ to $true$, and $\perp$ to $\perp$. We use $x <^3 y$ (for all $x, y \in \{true, \perp, false\}$) to mean that $x \leq^3 y$ and $x \neq y$.

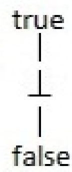


Figure 4.1: Truth Ordering $\leq^3$

From above, we can see that $\mathcal{M} = (\mathcal{L}, \neg^3)$, where $\mathcal{L} = (\{true, false, \bot\}, \leq^3) = (\{true, false, \bot\}, \leq^3, \vee^3, \wedge^3, false, true)$, is a multi-valued structure. Therefore, by interpreting the syntax of 3-valued ALFP over $\mathcal{M}$, we can derive the semantics of 3-valued ALFP and all theoretical results developed for multi-valued ALFP in Section 4.2.1 are preserved in the 3-valued setting. Since $\mathcal{M}$ here is also finite and distributive, we know from Section 4.3 that 3-valued ALFP can be reduced to 2-valued ALFP as well.

Let $Int^3 : \prod_k Rel_k \rightarrow \mathcal{U}^k \rightarrow \{true, \bot, false\}$ be a mapping. We define the 3-valued interpretation of ALFP in Table 4.4 where $\varrho \in Int^3$ maps each k-ary predicate symbol R to a 3-valued function and σ is an interpretation of variables. Notice that the truth value of $[(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ pre]$ is three valued, but the truth value of $[(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ cl]$ still remains two valued. Given σ_0 and a clause cl, a mapping ϱ satisfies cl if and only if $[(\varrho, \sigma_0)\ \underline{\mathtt{sat}}^3\ cl] = true$.

$$
\begin{array}{lcl}
[(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ R(v_1, ..., v_n)] & = & \varrho(R)(\sigma(v_1), ..., \sigma(v_n)) \\
[(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ \neg R(v_1, ..., v_n)] & = & \neg^3[(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ R(v_1, ..., v_n)] \\
[(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ pre_1 \wedge pre_2] & = & [(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ pre_1]\ \wedge^3 [(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ pre_2] \\
[(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ pre_1 \vee pre_2] & = & [(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ pre_1]\ \vee^3 [(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ pre_2] \\
[(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ \forall x:\ pre] & = & \min_{a \in \mathcal{U}}\{[(\varrho, \sigma[x \mapsto a])\ \underline{\mathtt{sat}}^3\ pre]\} \\
[(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ \exists x:\ pre] & = & \max_{a \in \mathcal{U}}\{[(\varrho, \sigma[x \mapsto a])\ \underline{\mathtt{sat}}^3\ pre]\} \\
\hline
[(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ \mathbf{true}] & = & \mathbf{true} \\
[(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ cl_1 \wedge cl_2] & = & [(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ cl_1]\ \wedge [(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ cl_2] \\
[(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ pre \Rightarrow R(v_1, ..., v_n)] & = & \begin{cases} true & [(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ pre] \leq^3 [(\varrho, \sigma) \\ & \underline{\mathtt{sat}}^3\ R(v_1, ..., v_n)] \\ false & \mathtt{otherwise} \end{cases} \\
[(\varrho, \sigma)\ \underline{\mathtt{sat}}^3\ \forall x:\ cl] & = & \forall a \in \mathcal{U} : [(\varrho, \sigma[x \mapsto a])\ \underline{\mathtt{sat}}^3\ cl] = true
\end{array}
$$

Table 4.4: Three-valued Interpretation of ALFP

Let us consider the mappings $S, S_1, S_2 : \mathcal{U}^k \to \{true, \perp, false\}$. For the truth ordering $\leq^3$, we have the following definitions. We define that $S_1 \leq^3 S_2$ iff $\forall x \in \mathcal{U}^k : S_1(x) \leq^3 S_2(x)$. Given an index set I, the greatest lower bound is defined as $S = \bigwedge^3_{i \in I} S_i$ iff $\forall x \in \mathcal{U}^k : S(x) = \wedge^3_{i \in I} S_i(x)$. We write $<^3$ for the irreflexive part of $\leq^3$.

The lexicographic ordering $\leq^3_\sharp$ for the interpretations of relations is defined as follows: $\varrho_1 \leq^3_\sharp \varrho_2$ if there exists a rank $i \in \{0, ..., r\}$ for a stratified clause $cl = \bigwedge_{0 \leq i \leq r} cl_i$ such that

- $\varrho_1(R) = \varrho_2(R)$ whenever $\texttt{rank}_R < i$,
- $\varrho_1(R) \leq^3 \varrho_2(R)$ whenever $\texttt{rank}_R = i$,
- either $i = r$ or $\varrho_1(R) <^3 \varrho_2(R)$ for some R with $\texttt{rank}_R = i$.

We also define $\varrho_1 \leq^3 \varrho_2$ to mean that $\varrho_1(R) \leq^3 \varrho_2(R)$ for all $R \in \mathcal{R}$.

The existence of the least model of 3-valued interpretations is guaranteed by the following corollary.

Corollary 4.13 *$\{\varrho | [(\varrho, \sigma_0)\ \underline{\boldsymbol{sat}}^3\ cl] = true\}$ is a Moore Family with respect to truth ordering, i.e. is closed under greatest lower bounds, whenever cl is closed and stratified; the greatest lower bound $\wedge^3_\sharp\ \{\varrho | [(\varrho, \sigma_0)\ \underline{\boldsymbol{sat}}^3\ cl] = true\}$ is the least model of cl.*

More generally, given ϱ_0 the set $\{\varrho | [(\varrho, \sigma_0)\ \underline{\boldsymbol{sat}}^3\ cl] = true \wedge \varrho_0 \leq^3 \varrho\}$ is a Moore Family with respect to truth ordering and $\wedge^3_\sharp\ \{\varrho | [(\varrho, \sigma_0)\ \underline{\boldsymbol{sat}}^3\ cl] = true \wedge \varrho_0 \leq^3 \varrho\}$ is the least model.

Proof. It's obvious from Theorem 4.2. □

4.5.3 Three-valued CTL

In this section, we introduce 3-valued CTL briefly. We consider the following fragment of CTL where formulas ϕ over a set of propositions $\mathbf{P}$ is defined as follows:

$$\phi ::= \ \mathbf{true} \mid p \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \mathbf{EX}\phi \mid \mathbf{E}[\phi_1 \mathbf{U} \phi_2] \mid \mathbf{AF}\phi$$

where $p \in \mathbf{P}$.

The semantics for 3-valued CTL formulas with respect to Kripke MTSs is defined in Table 4.5. This definition is obtained from the one in [41] by using the equivalence $\mathbf{AF}\phi \equiv \mathbf{A}[\mathbf{true}\mathbf{U}\phi]$ to obtain the semantics of the **AF** operator. One can check that the above equivalence and those listed in the following hold according to [41].

$$\begin{array}{lcl} \mathbf{AX}\phi & \equiv & \neg\mathbf{EX}\neg\phi \\ \mathbf{EF}\phi & \equiv & \mathbf{E}[\mathbf{true}\mathbf{U}\phi] \\ \mathbf{A}[\phi_1\mathbf{U}\phi_2] & \equiv & \neg\mathbf{E}[\neg\phi_2\mathbf{U}(\neg\phi_1 \wedge \neg\phi_2)] \wedge \mathbf{AF}\phi_2 \end{array}$$

An equivalent semantics for the temporal operators **EX**, **EU** and **AF** is given in Table 4.6, where we focus on the following two cases. One case is when a temporal formula ϕ evaluates to *true* and the other is when the formula ϕ evaluates to either *true* or $\perp$ according to Table 4.5. The semantics defined in Table 4.6 helps to better understand the flow logic approach to the analysis of Kripke MTSs which will be developed in the next section. It is easy to verify that the 3-valued semantics of the temporal operators **EX**, **EU** and **AF** given in Table 4.5 and Table 4.6 are equivalent.

Remark: To help better understand the three-valued CTL, we provide a further explanation briefly. Due to the state explosion problem when modeling systems with concrete models, abstraction techniques have been used to build abstract models of systems which result in much smaller sizes. A Kripke MTS itself is a formalism of abstract models. The three-valued CTL introduced here can be used to reason Kripke MTSs. Assume that a Kripke MTS M_A is an abstraction of a (concrete) Kripke structure M_C and that M_A satisfies a CTL formula (which means all the initial states of M_A satisfy this formula). A natural question that arises is whether M_C satisfies this formula (which means all the initial states of M_C satisfy this formula) as well. The three-valued semantics of CTL introduced in this section guarantees that if M_A satisfies (resp. does not satisfies) a CTL formula ϕ, then M_C also satisfies (resp. does not satisfies) the formula ϕ. We explain this formally in the following.

We rephrase the definition of *mixed simulation* introduced in [41, 52, 40, 106]

$$
\begin{array}{lcl}
[(M,s) \models^3 \mathbf{true}] & = & true \\
[(M,s) \models^3 p] & = & \begin{cases} true & L(s,p) = true \\ false & L(s,p) = false \\ \bot & L(s,p) = \bot \end{cases} \\
[(M,s) \models^3 \neg\phi] & = & \neg^3[(M,s) \models^3 \phi] \\
[(M,s) \models^3 \phi_1 \vee \phi_2] & = & [(M,s) \models^3 \phi_1] \vee^3 [(M,s) \models^3 \phi_2] \\
[(M,s) \models^3 \phi_1 \wedge \phi_2] & = & [(M,s) \models^3 \phi_1] \wedge^3 [(M,s) \models^3 \phi_2] \\
[(M,s) \models^3 \mathbf{EX}\phi] & = & \begin{cases} true & \texttt{there exists a must path } \pi \texttt{ from} \\ & s \texttt{ such that } |\pi| > 1 \texttt{ and} \\ & [(M,\pi[1]) \models^3 \phi] = true \\ false & \texttt{if for each may path } \pi \texttt{ from } s, \\ & \texttt{we have } [(M,\pi[1]) \models^3 \phi] = false \\ \bot & \texttt{otherwise} \end{cases} \\
[(M,s) \models^3 \mathbf{E}[\phi_1 \mathbf{U} \phi_2]] & = & \begin{cases} true & \texttt{there exists a must path } \pi \texttt{ from} \\ & \text{s} \texttt{ such that } \exists\, 0 \le k < |\pi| : \\ & [(([(M,\pi[k]) \models^3 \phi_2] = true) \\ & \wedge\ (\forall\, 0 \le j < k : [(M,\pi[j]) \models^3 \phi_1] = \\ & true)] \\ false & \texttt{if for each may path } \pi \texttt{ from } s, \\ & \texttt{we know that } (\forall\, 0 \le k < |\pi| : \\ & [(\forall\, 0 \le j < k : [(M,\pi[j]) \models^3 \phi_1] \ne \\ & false) \Rightarrow ([(M,\pi[k]) \models^3 \phi_2] = \\ & false)]) \wedge ((\forall\, 0 \le k < |\pi| : \\ & [(M,\pi[k]) \models^3 \phi_1] \ne false) \Rightarrow |\pi| = \infty) \\ \bot & \texttt{otherwise} \end{cases} \\
[(M,s) \models^3 \mathbf{AF}\phi] & = & \begin{cases} true & \texttt{if for each may path } \pi \texttt{ from } s, \\ & \texttt{we know that } \exists\, 0 \le k < |\pi| : \\ & [(M,\pi[k]) \models^3 \phi] = true \\ false & \texttt{if there exists a must path } \pi \\ & \texttt{from } s \texttt{ such that } \forall\, 0 \le k < |\pi| : \\ & [(M,\pi[k]) \models^3 \phi] = false \wedge |\pi| = \infty \\ \bot & \texttt{otherwise} \end{cases}
\end{array}
$$

Table 4.5: Three-valued Semantics for CTL

in the following.

Definition 4.14 Let $M_C = (S_C, S_{0_C}, T, L_C)$ be a Kripke structure over

$$[(M,s) \models^3 \mathbf{EX}\phi] = \begin{cases} true & \texttt{there exists a must path } \pi \\ & \texttt{from } s \texttt{ such that } |\pi| > 1 \texttt{ and} \\ & [(M,\pi[1]) \models^3 \phi] = true \\ true \texttt{ or } \bot & \texttt{there exists a may path } \pi \\ & \texttt{from } s \texttt{ such that} \\ & [(M,\pi[1]) \models^3 \phi] \neq false \\ false & \texttt{otherwise} \end{cases}$$

$$[(M,s) \models^3 \mathbf{E}[\phi_1 \mathbf{U} \phi_2]] = \begin{cases} true & \texttt{there exists a must path } \pi \\ & \texttt{from } s \texttt{ such that } \exists\, 0 \leq k < |\pi| : \\ & [([(M,\pi[k]) \models^3 \phi_2] = true) \\ & \wedge\ (\forall\, 0 \leq j < k : [(M,\pi[j]) \models^3 \phi_1] \\ & = true)] \\ true \texttt{ or } \bot & \texttt{there exists a may path } \pi \\ & \texttt{from } s \texttt{ such that } \exists\, 0 \leq k < |\pi| : \\ & [([(M,\pi[k]) \models^3 \phi_2] \neq false) \\ & \wedge\ (\forall\, 0 \leq j < k : [(M,\pi[j]) \models^3 \phi_1] \\ & \neq false)] \\ false & \texttt{otherwise} \end{cases}$$

$$[(M,s) \models^3 \mathbf{AF}\phi] = \begin{cases} true & \texttt{if for each may path } \pi \texttt{ from} \\ & s\texttt{, we know that } \exists\, 0 \leq k < |\pi| : \\ & [(M,\pi[k]) \models^3 \phi] = true \\ true \texttt{ or } \bot & \texttt{if for each must path } \pi \texttt{ from} \\ & s\texttt{, we know either } |\pi| \neq \infty \texttt{ or} \\ & \exists\, 0 \leq k < |\pi| : [(M,\pi[k]) \models^3 \phi] \\ & \neq false \\ false & \texttt{otherwise} \end{cases}$$

Table 4.6: Three-valued Semantics for Temporal Operators in CTL

atomic propositions set P, and let $M_A = (S_A, S_{0_A}, \xrightarrow{must}, \xrightarrow{may}, L_A)$ be an abstract Kripke MTS over P. We say that $H \subseteq S_C \times S_A$ is a *mixed simulation* from M_C to M_A if $(s_c, s_a) \in H$ implies the following:

1. $\forall p \in P$: if $p \in L_A(s_a)$ (resp. $p \notin L_A(s_a)$), then $L_C(s_c, p) = true$ (resp. $L_C(s_c, p) = false$).
2. if $s_c \to s'_c$, then there is some $s'_a \in S_A$ such that $s_a \xrightarrow{may} s'_a$ and $(s'_c, s'_a) \in H$.
3. if $s_a \xrightarrow{must} s'_a$, then there is some $s'_c \in S_C$ such that $s_c \to s'_c$ and $(s'_c, s'_a) \in H$.

We say that M_A is an abstraction of M_C (or M_C is represented by M_A), denoted $M_C \preceq M_A$, if we have a mixed simulation H such that $\forall s_c \in S_{0_C}, \exists s_a \in S_{0_A}$: $(s_c, s_a) \in H$ and $\forall s_a \in S_{0_A}, \exists s_c \in S_{0_C} : (s_c, s_a) \in H$.

We define $[M \models^3 \phi] = true$ (resp. $[M \models^3 \phi] = false$) to mean that $\forall s_0 \in S_0 : [(M, s_0) \models^3 \phi] = true$ (resp. $[(M, s_0) \models^3 \phi] = false$). Otherwise, $[M \models^3 \phi] = \perp$. We define that $[M \models \phi] = true$ iff $(M, s) \models \phi$ and that $[M \models \phi] = false$ iff $(M, s) \not\models \phi$.

Information ordering $\sqsubseteq^3$, depicted in Figure 4.2 on truth values is defined by $\perp \sqsubseteq^3 true$, $\perp \sqsubseteq^3 false$, $x \sqsubseteq^3 x$ (for all $x \in \{true, \perp, false\}$), and $x \not\sqsubseteq y$ otherwise.

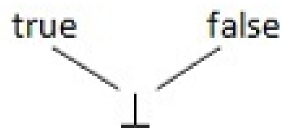


Figure 4.2: Information Ordering $\sqsubseteq^3$

The following theorem guarantees that if a Kripke MTS M_A satisfies (resp. does not satisfies) a CTL formula ϕ, then for a Kripke structure M_C represented by M_A, we have that M_C also satisfies (resp. does not satisfies) the formula ϕ. This helps to understand the three-valued semantics of CTL introduced in this section.

Theorem 4.15 *[41, 40] Let $H \subseteq S_C \times S_A$ be a mixed simulation relation from a concrete Kripke structure M_C to a Kripke MTS M_A. Then, for each $s_c \in S_C$ and $s_a \in S_A$ such that $(s_c, s_a) \in H$ and every CTL formula ϕ, we have that $[(M_A, s_a) \models^3 \phi] \sqsubseteq^3 [(M_C, s_c) \models \phi]$. Moreover, when $M_C \preceq M_A$, for every CTL formula ϕ, we have $[M_A \models^3 \phi] \sqsubseteq^3 [M_C \models \phi]$.*

Example 4.4 *Let $M_C = (S_C, S_{0_C}, T, L_C)$ be a concrete Kripke structure atomic propositions set P and $M_A = (S_A, S_{0_A}, \xrightarrow{must}, \xrightarrow{may}, L_A)$ be an abstract*

Kripke MTS over P. Let $\boldsymbol{EX}p$ be a CTL formula, where p is an atomic proposition. Assume that $s_c \in S_C$ and $s_a \in S_A$ such that $(s_c, s_a) \in H$, where H is a mixed simulation relation from M_C to M_A.

Assume that $[(M_A, s_a) \models^3 \boldsymbol{EX}p] = true$. According to the semantics of three-valued CTL, we know that $\exists s'_a \in S_A : s_a \xrightarrow{must} s'_a \wedge [(M_A, s'_a) \models^3 p] = true$. From Definition 4.14, we know that there is some $s'_c \in S_C$ such that $s_c \rightarrow s'_c$ and $(s'_c, s'_a) \in H$. Since $L_A(s'_a, p) \sqsubseteq^3 L_C(s'_c, p)$, we know that $[(M_C, s'_c) \models p] = true$. Therefore, from two-valued CTL semantics, we know that $[(M_C, s_c) \models \boldsymbol{EX}p] = true$.

Assume that $[(M_A, s_a) \models^3 \boldsymbol{EX}p] = false$. According to the semantics of three-valued CTL, we know that $\forall s'_a \in S_A$ such that $s_a \xrightarrow{may} s'_a$, $[(M_A, s'_a) \models^3 p] = false$. Let s'_c be a state in S_C such that $s_c \rightarrow s'_c$. From Definition 4.14, we know that there is some $s'_a \in S_A$ such that $s_a \xrightarrow{may} s'_a$ and $(s'_c, s'_a) \in H$. Since $L_A(s'_a, p) \sqsubseteq^3 L_C(s'_c, p)$, we know that $[(M_C, s'_c) \models p] = false$. Therefore, from two-valued CTL semantics, we know that $[(M_C, s_c) \models \boldsymbol{EX}p] = false$.

From above, we know that $[M_A \models^3 \boldsymbol{EX}p] \sqsubseteq^3 [M_C \models \boldsymbol{EX}p]$.

4.5.4 Three-valued CTL in Three-valued ALFP

In this section, we use three-valued ALFP to analyze Kripke MTSs. It has been pointed out in Section 4.4 that the flow logic approach developed in Table 3.1 naturally generalizes to a multi-valued analysis of CTL over a multi-valued TS when using multi-valued ALFP to interpret those ALFP clauses. (Our multi-valued analysis for CTL has been listed in Table 4.3, where we have also made a necessary modification in order to analyze the formula **true** using multi-valued ALFP.) As an application of this observation, we focus on the 3-valued setting. By interpreting those ALFP constraints over 3-valued ALFP semantics, we get a 3-valued analysis for 3-valued CTL over a Kripke MTSs. Moreover, a stronger result is provided in this section, that is 3-valued ALFP could encode 3-valued CTL model checking over Kripke MTSs.

To encode a Kripke MTS $(S, S_0, \xrightarrow{must}, \xrightarrow{may}, L)$ into 3-valued ALFP, we can define corresponding predicates in ϱ_0 as follows. The universe $\mathcal{U} = S$.

- for each atomic proposition p over $\mathbf{P}$, we define a predicate P_p such that

$\varrho_0(P_p)(s) = L(s,p)$,

- we define a transition relation T such that $\varrho_0(T)(s,s') = \mathit{true}$ if $(s,s') \in \overset{must}{\longrightarrow}$, $\varrho_0(T)(s,s') = \perp$ if $(s,s') \in \overset{may}{\longrightarrow}$ but $(s,s') \notin \overset{must}{\longrightarrow}$, and $\varrho_0(T)(s,s') = \mathit{false}$ otherwise.

We explain Table 4.3 in three-valued setting in the following. For each CTL formula ϕ, there is a judgement of the form $\vec{R} \vdash \phi$ to define a relation R_ϕ. The intention is that $[(M,s) \models^3 \phi] = \varrho(R_\phi)(s)$ holds in the least model ϱ satisfying $\vec{R} \vdash \phi \wedge \varrho_0 \leq^3 \varrho$.

For the relation R_{true} corresponding to the CTL formula **true**, $\varrho(R_{true})$ should map each state s to true and this is guaranteed by the ALFP clause $\forall s : \mathit{True}(s) \Rightarrow R_{true}(s)$. For the atomic proposition p we make use of the pre-defined predicate P_p and impose the constraint $\forall s : P_p(s) \Rightarrow R_p(s)$ such that $\varrho(R_p)$ maps a state s to the same truth value as $\varrho(P_p)$ does. The clauses for boolean operators $\vee, \wedge$ and $\neg$ follow the same pattern so we just explain one of them, namely disjunction $\phi_1 \vee \phi_2$. The judgements $\vec{R} \vdash \phi_1$ and $\vec{R} \vdash \phi_2$ ensure that for the relations $R_{\phi'}$ corresponding to subformulas of ϕ_1 or ϕ_2, $\varrho(R_{\phi'})$ map states to truth values correctly. The clause $\forall s : R_{\phi_1}(s) \vee R_{\phi_2}(s) \Rightarrow R_{\phi_1 \vee \phi_2}(s)$ requires that $R_{\phi_1 \vee \phi_2}(s)$ is mapped to true (resp. true or $\perp$) if $R_{\phi_1}(s)$ or $R_{\phi_2}(s)$ is mapped to true (resp. true or $\perp$).

In the case of $\mathbf{EX}\phi$, the first conjunct ensures that for the relations $R_{\phi'}$ corresponding to subformulas of ϕ, $\varrho(R_{\phi'})$ map states to truth values correctly. The second conjunct requires that if there is a *must* (resp. *may*) transition from s to s', i.e. $\varrho(T)(s,s')$ equals to true (resp. true or $\perp$), and $R_\phi(s')$ is mapped to true (resp. true or $\perp$), then $R_{\mathbf{EX}\phi}(s)$ is mapped to true (resp. true or $\perp$). The above case corresponds to the true (resp. true or $\perp$) case in the semantics of the **EX** operator in Table 4.6.

The clause for $\mathbf{E}[\phi_1 \mathbf{U} \phi_2]$ captures two possibilities. If $R_{\phi_2}(s)$ is mapped to true (resp. true or $\perp$), then $\varrho(R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]})$ should map s to true (resp. true or $\perp$). Alternatively if $R_{\phi_1}(s)$ is mapped to true (resp. true or $\perp$) and there is a *must* (resp. *may*) transition from s to s', i.e. $\varrho(T)(s,s')$ equals to true (resp. true or $\perp$), and $R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}(s')$ is mapped to true (resp. true or $\perp$), then $\varrho(R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]})$ should also map s to true (resp. true or $\perp$).

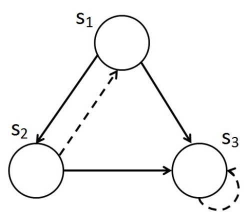
We pay slightly more attention to the clause for $\mathbf{AF}\phi$ due to the existence of stuck states with respect to *must* transitions. If $R_\phi(s)$ is mapped to *true* (resp. *true* or $\perp$), then $\varrho(R_{\mathbf{AF}\phi})$ should map s to *true* (resp. *true* or $\perp$). If $\varrho(R_{\mathbf{AF}\phi})$ maps all *may* (resp. *must*) successors s' of s, i.e. $\varrho(T)(s,s')$ equals to *true* or $\perp$ (resp. *true*), to *true* (resp. *true* or $\perp$), then we impose that $R_{\mathbf{AF}\phi}(s)$ is mapped to *true* (resp. *true* or $\perp$). Notice that if there are no outgoing *must* transitions from s, then $\neg\varrho(T)(s,s') = true$ or $\perp$ for any state s'. In this case, the third conjunct requires $\varrho(R_{\mathbf{AF}\phi})$ to map s to *true* or $\perp$.

We have the following theorem saying that the best analysis result of our flow logic approach to the analysis of Kripke MTSs coincides with the solutions for the model checking problem for 3-valued CTL with respect to Kripke MTSs.

Theorem 4.16 *For a CTL formula ϕ and the least model ϱ of $\vec{R} \vdash \phi$ such that $\varrho = \wedge^3_\sharp\{\varrho \mid [(\varrho,\sigma)\ \underline{\boldsymbol{sat}}^3\ (\vec{R} \vdash \phi)] = true,\ \varrho_0 \leq^3 \varrho\}$, where ϱ_0 defines P_p, T and $True$, we know that $[(M,s) \models^3 \phi] = \varrho(R_\phi)(s)$.*

Proof. In Appendix B. □

Example 4.5 *Consider a Kripke MTS, given by the diagram to the left, with $S = \{s_1,s_2,s_3\}$, $S_0 = \{s_1\}$, $L(s_1,p) = L(s_2,p) = false$ and $L(s_3,p) = true$. Solid lines represent transitions in $\overset{must}{\longrightarrow}$ and dashed lines denote transitions in $\overset{may}{\longrightarrow} \setminus \overset{must}{\longrightarrow}$.*



s	$\varrho(R_{\boldsymbol{AF}p})(s)$	$[(M,s) \models^3 \boldsymbol{AF}p]$
s_1	$\perp$	$\perp$
s_2	$\perp$	$\perp$
s_3	*true*	*true*

We evaluate the CTL formula $\boldsymbol{AF}p$ over the above Kripke MTS using the 3-valued ALFP and the 3-valued semantics of CTL respectively. The results are given in the table to the right. We can see that model checking and our static analysis give the same result.

Using our static analysis approach, we first encode the above Kripke MTS in ϱ_0 *and then specify our analysis with the judgement* $\vec{R} \vdash \boldsymbol{AF}p$*. According to Table 4.3, the following clause cl*

$$
\begin{array}{l}
[\forall s : P_p(s) \Rightarrow R_p(s)]\wedge \\
[\forall s : R_p(s) \Rightarrow R_{\boldsymbol{AF}p}(s)]\wedge \\
[\forall s : [\forall s' : \neg T(s, s') \vee R_{\boldsymbol{AF}p}(s')] \Rightarrow R_{\boldsymbol{AF}p}(s)]
\end{array}
$$

will be generated and the least solution ϱ *to cl subject to* $\varrho_0 \leq^3 \varrho$ *can then be calculated according to the 3-valued semantics of ALFP.*

4.6 Future Work

In this chapter, we have developed a multi-valued analysis for CTL (without fairness). In our future work, we are interested in comparing our multi-valued analysis result with the semantics of multi-valued CTL proposed in [43]. In two-valued model checking, fairness assumptions have been used to rule out unrealistic computation paths. We are also interested in introducing fairness assumptions into the multi-valued setting and developing a multi-valued analysis for CTL with fairness assumptions.

Chapter 5

Alternation-free μ-calculus in Alternation-free Least Fixed Point Logic

Chapter 3 presents a flow logic approach to static analysis which encodes model checking of CTL formulas in ALFP. In this chapter, we continue the line of work there and focus on a larger fragment of temporal logic, namely the *Alternation-free fragment of the μ-calculus*, and show that this fragment of logic can be characterised in a similar way. To do this, we first propose an Alternation-free Normal Form (AFNF), where negations are only applied to closed subformulas; the expressive power of closed formulas in AFNF is equivalent to the alternation-free fragment of the μ-calculus. Then, we show that model checking for the alternation-free μ-calculus can be encoded in ALFP with the usual notion of *stratification*, i.e. the Moore family result makes use of a lexicographic ordering imposed by a suitable choice of ranking of the relations in the ALFP formula.

When negations are applied to open μ-calculus subformulas, our encoding method fails. We therefore establish a negative result showing that there exists a μ-calculus formula of alternation depth 2 whose least fixed point semantics cannot be characterized as a Moore Family property in ALFP with respect to any notion of ranking.

The structure of this chapter is as follows. In Section 5.1, we introduce the alternation-free fragment of the modal μ-calculus. First, we give the definition of alternation depth of the μ-calculus and define the alternation-free μ-calculus. Then we propose Alternation-free Normal Form. The encoding of the alternation-free μ-calculus into ALFP is introduced in Section 5.2. Section 5.3 explains our negative result.

5.1 The Alternation-free Fragment of the Modal μ-calculus

5.1.1 The Alternation Depth of the μ-calculus

Definitions of the alternation depth for modal μ-calculus formulas can be found in [6, 7, 8]. Based on [8], where the definition of the alternation depth is given for a version of the modal μ-calculus with simultaneous fixpoints, we give our definition for the modal μ-calculus with just unary fixpoints.

We say that a formula φ is a *proper* subformula of formula ϕ iff φ is a subformula of ϕ but is not ϕ itself. A formula is called a μ-formula iff its main connective is μ. A subformula φ of ϕ is called a μ-subformula of it iff the main connective of φ is μ. The notions of ν-formula and ν-subformula can be defined similarly. Both μ-formula and ν-formula are called fixpoint formula, and similarly μ-subformula and ν-subformula are called fixpoint subformula. A μ-subformula φ of ϕ is called a *top-level* μ-subformula of it iff φ is not a μ-subformula of any other μ-subformula of ϕ. A μ-subformula φ of ϕ is called a *top* μ-subformula of it iff φ is not a μ-subformula of any other fixpoint subformula of ϕ. The notions of *top-level* ν-subformula and *top* ν-subformula can be defined similarly. Given a set of μ-calculus formulas, a formula in the set is called a *maximal* formula of the set iff it is not a proper subformula of any other formulas in this set.

Definition 5.1 (The Alternation Depth of Formulas) For a closed μ-calculus formula ϕ given in Negation-free PNF, the alternation depth, $ad(\phi)$, is defined inductively as follows (assuming that $max\{\emptyset\} = 0$).

1. If ϕ contains closed proper fixpoint subformulas, and ϕ_1,...,ϕ_n are the maximal formulas of the set of closed proper fixpoint subformulas of ϕ,

then

$$ad(\phi) = max\{ad(\phi'), ad(\phi_1), ..., ad(\phi_n)\}$$

where ϕ' is obtained from ϕ by substituting new atomic propositions p_1,...,p_n for ϕ_1,...,ϕ_n.

2. If ϕ contains no closed proper fixpoint subformulas then $ad(\phi)$ is defined as follows.

   - $ad(p) = 0$, for any atomic proposition p.
   - $ad(\phi_1 \vee \phi_2) = ad(\phi_1 \wedge \phi_2) = max(ad(\phi_1),\ ad(\phi_2))$.
   - $ad([a]\varphi) = ad(\langle a\rangle\varphi) = ad(\varphi)$, for any transition relation a.
   - $ad(\mu Q.\varphi) = 1 + max\{ad(\varphi_1'), ..., ad(\varphi_n')\}$ where $\varphi_i (1 \leq i \leq n)$ is top-level ν-subformula of φ and $\varphi_i' (1 \leq i \leq n)$ is constructed from φ_i by substituting all free variables with any new propositions.
   - $ad(\nu Q.\varphi) = 1 + max\{ad(\varphi_1'), ..., ad(\varphi_n')\}$ where $\varphi_i (1 \leq i \leq n)$ is top-level μ-subformula of φ and $\varphi_i' (1 \leq i \leq n)$ is constructed from φ_i by substituting all free variables with any new propositions.

As in [7], we define the *alternation-free* fragment of the μ-calculus formulas as those formulas whose alternation depth are zero or one.

Example 5.1 *Let $\phi = \nu Q_1.((p \wedge \langle a\rangle Q_1) \vee \mu Q_2.(q \wedge \langle a\rangle Q_2))$ be a μ-calculus formula. We can see that ϕ contains a closed proper fixpoint subformula $\phi_1 = \mu Q_2.(q \wedge \langle a\rangle Q_2)$. We substitute the subformula ϕ_1 in ϕ with p_1 and we get $\phi' = \nu Q_1.((p \wedge \langle a\rangle Q_1) \vee p_1)$. According to Definition 5.1, we know that $ad(\phi) = max\{ad(\phi'), ad(\phi_1)\}$. Since ϕ' contains no closed proper fixpoint subformulas and it contains no top-level μ-subformulas, we know that $ad(\phi') = 1$. Since ϕ_1 contains no closed proper fixpoint subformulas and it contains no top-level ν-subformulas, we know that $ad(\phi_1) = 1$. Therefore, $ad(\phi) = max\{1, 1\} = 1$. Hence, ϕ is an alternation-free formula.*

5.1.2 Alternation-free Normal Form

In this section, we propose an Alternation-free Normal Form (AFNF) and show that closed formulas in AFNF exactly characterize the alternation-free fragment of the modal μ-calculus. This will facilitate our subsequent development.

Definition 5.2 (Syntax of Alternation-Free Normal Form) Let Var be a set of variables, $\mathbf{P}$ be a set of atomic propositions that is closed under negation. The syntax of Alternation-free Normal Form is defined as follows:

$$\phi ::= \; p \mid Q \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid \langle a \rangle \phi \mid [a]\phi \mid \mu Q.\phi \mid \neg \mu Q.\phi$$

where no variable is quantified twice and $\neg \mu Q.\phi$ is a closed formula.

We focus on closed formulas in AFNF. In the following, we briefly show that close formulas in AFNF has the same expressive power as the alternation-free fragment of the modal μ-calculus. First we will show that all alternation-free μ-calculus formulas can be put in AFNF and the resulting formulas in AFNF are closed. Second we will show that all closed formulas in AFNF are indeed alternation-free.

We first have the following lemma about the alternation-free μ-calculus, which gives us some useful insights when proving Lemma 5.4.

Lemma 5.3 *For any alternation-free μ-calculus formula ϕ in Negation-free PNF, we have the following:*

1. *For any μ-subformula φ of ϕ, all top-level ν-subformulas of φ are closed.*
2. *For any ν-subformula φ of ϕ, all top-level μ-subformulas of φ are closed.*

Proof. We prove by contradiction. Assume that there exists an open top-level ν-subformula φ_1 for a μ-subformula φ of ϕ. According to Definition 5.1, $ad(\phi) \geq 1 + ad(\varphi_1') \geq 1 + 1 + max\{...\} \geq 2$. Therefore ϕ is not alternation-free and this contradicts our assumption. The proof for any ν-subformula of ϕ is similar. □

Translating Alternation-free μ-calculus to its Alternation-free Normal Form: Informally, we can use the following three steps to translate an alternation-free μ-calculus formula in Negation-free PNF to its Alternation-free Normal Form.

1. First, we use the duality $\nu Q.\phi \equiv \neg\mu Q.\neg\phi[\neg Q/Q]$ to eliminate all ν operators in the formula.
2. Second, we use De Morgan's law and the dualities $\neg[a]\phi \equiv \langle a\rangle\neg\phi$ and $\neg\langle a\rangle\phi \equiv [a]\neg\phi$ to push negations as deep as possible. When a negation is pushed to a positive occurrence of an atomic proposition, it cannot be pushed any deeper. Negated occurrences of atomic propositions might appear when this step is finished.
3. Finally, we substitute each negated occurrence $\neg p$ of atomic proposition p with a new atomic proposition p' to eliminate negations in front of atomic propositions.

Based on the above mentioned translation method, we have the following lemma.

Lemma 5.4 *Let ϕ be an alternation-free μ-calculus formula in Negation-free PNF and assume that we translate ϕ to its Alternation-free Normal Form ϕ' using our translation method. Then, each subformula of the form $\neg\mu Q.\varphi$ in the formula ϕ' is indeed closed and no negations are applied to variables in ϕ'.*

Proof. In Appendix C. □

Hence, we have the following, which finishes our proofs for one direction.

Lemma 5.5 *Every alternation-free μ-calculus formula ϕ in Negation-free PNF can be translated to its Alternation-free Normal Form ϕ' while preserving the semantics. The resulting formula ϕ' is closed.*

Proof. From Lemma 5.4, we know that after translating an alternation-free μ-calculus formula ϕ in negation-free PNF using our three-steps transformation, the formula ϕ' is indeed in Alternation-free Normal Form. It's obvious that ϕ' is closed. □

Example 5.2 *Let $\phi = \mu Q_1.((p \wedge \langle a\rangle Q_1) \vee \nu Q_2.(q \wedge \langle a\rangle Q_2))$ be an alternation-free μ-calculus formula in Negation-free PNF. We can translate ϕ to its equivalent Alternation-free Normal Form $\phi' = \mu Q_1.((p \wedge \langle a\rangle Q_1) \vee \neg\mu Q_2.(q' \vee [a]Q_2))$ where $q' \equiv \neg q$. We can see that ϕ' is closed.*

We now start to show the other direction.

Translating Alternation-free Normal Form to Negation-free PNF: By using the following three steps repeatedly, we can translate a formula ϕ in AFNF to its Negation-free PNF ϕ'.

1. First, we eliminate all maximal subformulas of the form $\neg\mu Q.\varphi$ in ϕ by duality $\neg\mu Q.\varphi \equiv \nu Q.\neg\varphi[\neg Q/Q]$.
2. Second, we use De Morgan's law and dualities $\neg[a]\varphi \equiv \langle a\rangle\neg\varphi$ and $\neg\langle a\rangle\varphi \equiv [a]\neg\varphi$ to push negations as deep as possible. Negated occurrences of atomic propositions might appear when this step is finished.
3. Third, we substitute each negated occurrence $\neg p$ of atomic proposition p with a new atomic proposition p' to eliminate negations in front of atomic propositions.

Notice that after we eliminate a subformula $\neg\mu Q.\varphi$ using the first step above, negations might be pushed to some positive occurrences of μ-subformulas of φ in the second step. Therefore, new negative occurrences of μ operators appear. We can go back and start from the first step again to eliminate newly occurred negative μ operators. Since each formula only has finite number of subformulas, only finite number of new negative occurrences of μ operators can appear. Therefore, this repetition will terminate and finally we can get the formula ϕ' in Negation-free PNF. The translation clearly preserves the semantics.

We have the following lemmas, which finishes the proof for the other direction.

Lemma 5.6 *Given a μ-calculus formula ϕ' in Negation-free PNF which is translated from a closed formula ϕ in Alternation-free Normal Form. Assume that $\phi'_1,\ldots,\phi'_n$ are the maximal formulas of the set of closed proper fixpoint subformulas of ϕ', the alternation depth of ϕ'', which is obtained from ϕ' by substituting new atomic propositions $p_1,\ldots,p_n$ for $\phi'_1,\ldots,\phi'_n$, is strict less than 2.*

Proof. In Appendix C. □

Lemma 5.7 *Every μ-calculus formula ϕ' in Negation-free PNF translated from a closed formula ϕ in Alternation-free Normal Form is alternation-free.*

PROOF. In Appendix C. □

EXAMPLE 5.3 *Let $\phi = \mu Q_1.((p \wedge \langle a \rangle Q_1) \vee \neg \mu Q_2.(q \vee \langle a \rangle Q_2))$ be a closed formula in Alternation-free Normal Form. We can translate ϕ to its equivalent Negation-free PNF $\phi' = \mu Q_1.((p \wedge \langle a \rangle Q_1) \vee \nu Q_2.(q' \wedge [a] Q_2))$ where $q' \equiv \neg q$. We can see that ϕ' is alternation free according to Definition 5.1.*

From above, we have the following proposition, which is the main result of this section.

PROPOSITION 5.8 *Closed formulas defined in Alternation-free Normal Form exactly characterize the alternation-free fragment of modal μ-calculus formulas.*

PROOF. It is obvious from Lemma 5.5 and Lemma 5.7. □

5.2 The Alternation-free Fragment of the μ-Calculus in ALFP

We encode the model checking problem for the alternation-free μ-calculus into ALFP. According to Proposition 5.8, we use closed formulas defined in Alternation-free Normal Form to characterize the alternation-free fragment of the μ-calculus.

We first encode a Kripke structure $M = (S, T, L)$ into ALFP by defining corresponding relations as follows. Recall that in the model checking problem for the μ-calculus, the definition of Kripke structure is slighted different with the one given in Section 2.3. Here, T is a set of transition relations, and each element a in T is a transition relation and $a \subseteq S \times S$. Assume that the universe is $\mathcal{U} = S$,

- for each atomic proposition p we define a predicate P_p such that $\varrho_0(P_p)(s)$ if and only if $p \in L(s)$, and
- for each element a in T, we define a binary relation a such that $\varrho_0(T_a)(s, t)$ if and only if $(s, t) \in a$.

We are most interested in variables in a μ-calculus formula. Therefore, we define only relations for all variables that occur in a given formula. We first introduce the idea of *Strongly Benign Translation* as follows.

Definition 5.9 A Strongly Benign Translation is a translation from a μ-calculus formula ϕ to an ALFP clause cl such that we define a relation R_Q in cl iff Q is a variable in ϕ.

To develop a Strongly Benign Translation for the alternation-free fragment of the μ-calculus, for each μ-calculus formula ϕ, we map it to a pair $\langle cl_\phi, pre_\phi\rangle$, where cl_ϕ is an ALFP clause and pre_ϕ is a precondition in ALFP. We use $pre_\phi[s'/s]$ to denote a precondition resulting from pre_ϕ by substituting the free variable s in pre_ϕ with s'. Assume ϱ is the least model of cl_ϕ subject to $\varrho(R_{Q_1}) \supseteq S_1, ..., \varrho(R_{Q_n}) \supseteq S_n, \varrho \supseteq \varrho_0$, where ϱ_0 defines P_p and T_a and $Q_1, ..., Q_n$ are all the free variables in ϕ. The intention of our development is that $s' \in [\![\phi]\!]_{e[Q_1 \mapsto S_1,...,Q_n \mapsto S_n]}$ iff $(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_ϕ, and that when ϕ takes the form $\mu Q.\phi$, we have that $[\![\mu Q.\phi]\!]_{e[Q_1 \mapsto S_1,...,Q_n \mapsto S_n]}$ equals $\varrho(R_Q)$. The Strongly Benign Translation we have developed is given in Table 5.1.

$$
\begin{array}{lcl}
p & \longmapsto & \langle \mathbf{true},\ P_p(s)\rangle \\
Q & \longmapsto & \langle \mathbf{true},\ R_Q(s)\rangle \\
\phi_1 \vee \phi_2 & \longmapsto & \langle cl_{\phi_1} \wedge cl_{\phi_2},\ pre_{\phi_1} \vee pre_{\phi_2}\rangle \\
 & & \texttt{whenever}\ \phi_1 \longmapsto \langle cl_{\phi_1},\ pre_{\phi_1}\rangle\ \texttt{and}\ \phi_2 \longmapsto \langle cl_{\phi_2},\ pre_{\phi_2}\rangle \\
\phi_1 \wedge \phi_2 & \longmapsto & \langle cl_{\phi_1} \wedge cl_{\phi_2},\ pre_{\phi_1} \wedge pre_{\phi_2}\rangle \\
 & & \texttt{whenever}\ \phi_1 \longmapsto \langle cl_{\phi_1},\ pre_{\phi_1}\rangle\ \texttt{and}\ \phi_2 \longmapsto \langle cl_{\phi_2},\ pre_{\phi_2}\rangle \\
\langle a\rangle\phi & \longmapsto & \langle cl_\phi,\ \exists s' : T_a(s,s') \wedge pre_\phi[s'/s]\rangle \\
 & & \texttt{whenever}\ \phi \longmapsto \langle cl_\phi,\ pre_\phi\rangle \\
[a]\phi & \longmapsto & \langle cl_\phi,\ \forall s' : \neg T_a(s,s') \vee pre_\phi[s'/s]\rangle \\
 & & \texttt{whenever}\ \phi \longmapsto \langle cl_\phi,\ pre_\phi\rangle \\
\mu Q.\phi & \longmapsto & \langle [\forall s : pre_\phi \Rightarrow R_Q(s)] \wedge cl_\phi,\ R_Q(s)\rangle \\
 & & \texttt{whenever}\ \phi \longmapsto \langle cl_\phi,\ pre_\phi\rangle \\
\neg\mu Q.\phi & \longmapsto & \langle cl_{\mu Q.\phi},\ \neg R_Q(s)\rangle \\
 & & \texttt{whenever}\ \mu Q.\phi \longmapsto \langle cl_{\mu Q.\phi},\ pre_{\mu Q.\phi}\rangle
\end{array}
$$

Table 5.1: Strongly Benign Translation of the Alternation-free μ-calculus in ALFP

For atomic proposition p, we simply define cl_p as **true** since there are no bounded variables in p. We make use of the predefined predicate P_p and define pre_p as $P_p(s)$. For a variable Q, we also define cl_Q as **true** since the Q is a free variable here. We define pre_Q as $R_Q(s)$.

For $\phi_1 \vee \phi_2$, we assume that $\phi_1 \longmapsto \langle cl_{\phi_1},\ pre_{\phi_1}\rangle$ and $\phi_2 \longmapsto \langle cl_{\phi_2},\ pre_{\phi_2}\rangle$. This means that for each subformula $\mu Q.\phi$ in ϕ_1 (or ϕ_2), the relation R_Q is defined correctly in cl_{ϕ_1} (or cl_{ϕ_2}) and that pre_{ϕ_1} and pre_{ϕ_2} are also defined as expected. We define $cl_{\phi_1\vee\phi_2}$ as $cl_{\phi_1} \wedge cl_{\phi_2}$. This ensures that for each subformula $\mu Q.\phi$ in $\phi_1 \vee \phi_2$, R_Q is defined correctly in $cl_{\phi_1} \wedge cl_{\phi_2}$. It's also natural to define $pre_{\phi_1\vee\phi_2}$ as $pre_{\phi_1} \vee pre_{\phi_2}$. The case for $\phi_1 \wedge \phi_2$ follows the same pattern.

For $\langle a\rangle\phi$, we assume that $\phi \longmapsto \langle cl_\phi,\ pre_\phi\rangle$. This means that for each subformula $\mu Q.\varphi$ in ϕ, the relation R_Q is defined correctly in cl_ϕ and that pre_ϕ is also defined in an intended way. We simply define $cl_{\langle a\rangle\phi}$ to be the same as cl_ϕ since this suffices to guarantee that for each subformula $\mu Q.\varphi$ in $\langle a\rangle\phi$, the relation R_Q is defined correctly in $cl_{\langle a\rangle\phi}$. We define $pre_{\langle a\rangle\phi}$ as $\exists s' : T_a(s,s') \wedge pre_\phi[s'/s]$. This means for any state s if $pre_\phi[s'/s]$ holds on any of the a-derivative s' of s, then $pre_{\langle a\rangle\phi}$ holds on state s. This matches the semantics for $\langle a\rangle\phi$.

For $[a]\phi$, we also assume that $\phi \longmapsto \langle cl_\phi,\ pre_\phi\rangle$. For a similar reason as in the case for $\langle a\rangle\phi$, we define $cl_{[a]\phi}$ to be the same as cl_ϕ. We define $pre_{[a]\phi}$ as $\forall s' : \neg T_a(s,s') \vee pre_\phi[s'/s]$. This means for any state s if $pre_\phi[s'/s]$ holds on all of the a-derivatives s' of s, then $pre_{[a]\phi}$ holds on state s. Notice here that if s has no a-derivatives, $pre_{[a]\phi}$ still holds on s. This also matches the semantics for $[a]\phi$.

For $\mu Q.\phi$, we assume that $\phi \longmapsto \langle cl_\phi,\ pre_\phi\rangle$ as well. We define $cl_{\mu Q.\phi}$ as $[\forall s : pre_\phi \Rightarrow R_Q(s)] \wedge cl_\phi$. The first conjunct $[\forall s : pre_\phi \Rightarrow R_Q(s)]$ defines the relation R_Q and the second conjunct cl_ϕ ensures that for each proper subformula $\mu Q'.\varphi$ in ϕ, the relation $R_{Q'}$ is also defined correctly in cl_ϕ. The mapping here matches the semantics for the least fixed point operator μ. We define $pre_{\mu Q.\phi}$ as $R_Q(s)$.

For $\neg\mu Q.\phi$, we assume that $\mu Q.\phi \longmapsto \langle cl_{\mu Q.\phi},\ pre_{\mu Q.\phi}\rangle$. We define $cl_{\neg\mu Q.\phi}$ to be the same as $cl_{\mu Q.\phi}$. This guarantees that for each subformula $\mu Q'.\varphi$ in $\mu Q.\phi$, the relation $R_{Q'}$ is also defined correctly in $cl_{\neg\mu Q.\phi}$. We simple define $pre_{\neg\mu Q.\phi}$ as $\neg R_Q(s)$.

For a closed formula ϕ in AFNF, to show that the clause cl_ϕ is *stratified*, where $\phi \longmapsto \langle cl_\phi, pre_\phi\rangle$, we introduce a ranking method for cl_ϕ.

Assume that there are N variables in ϕ. For each variable Q that occurs in ϕ, the rank $\texttt{rank}_{R_Q}$ can be calculated according to the following steps.

1. If ϕ contains closed μ-subformulas and assume that $\mu Q_1.\phi_1, ..., \mu Q_n.\phi_n$ are the maximal formulas of the set of closed μ-subformulas of ϕ, we require that $\texttt{rank}_{R_{Q_i}} = N$ where $1 \leq i \leq n$. We add all these μ-subformulas $\mu Q_1.\phi_1, ..., \mu Q_n.\phi_n$ to a set Set, which is used to keep those unprocessed μ-subformulas.

2. For each of the μ-subformula $\mu Q_i.\phi_i$ in Set, assume that $\mu Q'_i.\phi'_i$ is a proper top μ-subformula of $\mu Q_i.\phi_i$. If $\mu Q'_i$ is a negative occurrence (negation is applied to $\mu Q'_i.\phi'_i$), then we require that $\texttt{rank}_{R_{Q'_i}} = \texttt{rank}_{R_{Q_i}} - 1$. If $\mu Q'_i$ is a positive occurrence (no negation is applied to $\mu Q'_i.\phi'_i$), we require that $\texttt{rank}_{R_{Q'_i}} = \texttt{rank}_{R_{Q_i}}$. We add all proper top μ-subformulas $\mu Q'_i.\phi'_i$ of $\mu Q_i.\phi_i$ to Set and remove $\mu Q_i.\phi_i$ from Set. We repeat the second step until Set becomes empty.

3. Assume that N' is the lowest rank of all the ranks that have been assigned to relations R_Qs when Set becomes empty. For each variable Q in ϕ, we modify $\texttt{rank}_{R_Q}$ by $\texttt{rank}_{R_Q} = \texttt{rank}_{R_Q} - (N' - 1)$. This makes sure that the lowest rank becomes 1.

We assign the predicate P_p and T_a the rank 0. The following lemma ensures stratification of our encoding.

Lemma 5.10 *Given a closed μ-calculus formula ϕ in AFNF and assume that $\phi \longmapsto \langle cl_\phi,\ pre_\phi \rangle$ according to Table 5.1, the clause cl_ϕ is closed and stratified.*

Proof. It's obvious from the above ranking method and Table 5.1. □

Example 5.4 *Let $\phi = \mu Q_1.(\mu Q_2.((\langle a \rangle Q_1 \vee \langle a \rangle Q_2) \wedge p) \vee \neg \mu Q_3.(q \vee \langle a \rangle Q_3))$ be a closed μ-calculus formula in AFNF. Assume that $\phi \longmapsto \langle cl_\phi,\ pre_\phi \rangle$ according to Table 5.1. According to our ranking method, we require that $\texttt{rank}_{R_{Q_1}} = \texttt{rank}_{R_{Q_2}} = 2$, $\texttt{rank}_{R_{Q_2}} = 1$ and $\texttt{rank}_{P_p} = \texttt{rank}_{P_q} = 0$. It is easy to see that cl_ϕ is stratified.*

The following theorem shows that the precondition pre_ϕ in our mapping $\phi \longmapsto \langle cl_\phi,\ pre_\phi\rangle$ correctly characterizes the semantics of ϕ.

Theorem 5.11 *Let ϕ be a μ-calculus formula in Alternation-free Normal Form with $Q_1, ..., Q_n$ being all the free variables in it. Assume that $\phi \longmapsto \langle cl_\phi,\ pre_\phi\rangle$. For the least solution ϱ of cl_ϕ such that $\varrho = \sqcap\{\varrho \mid (\varrho, \sigma)\ \underline{\textbf{sat}}\ cl_\phi \wedge \varrho(R_{Q_1}) \supseteq S_1, ..., \wedge\ \varrho(R_{Q_n}) \supseteq S_n \wedge \varrho \supseteq \varrho_0\}$, where ϱ_0 defines P_p and T_a, we have $s' \in [\![\phi]\!]_{e[Q_1 \mapsto S_1, ..., Q_n \mapsto S_n]}$ iff $(\varrho, \sigma[s \mapsto s'])\ \underline{\textbf{sat}}\ pre_\phi$.*

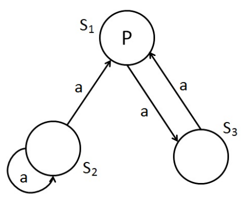
Proof. In Appendix C. □

We focus on alternation-free μ-calculus formulas of the form $\mu Q.\phi$. This is not a restriction since $[\![\phi]\!] = [\![\mu Q.\phi]\!]$ when Q is not a free variable in ϕ. From Theorem 5.11, we have the following corollary saying that the best analysis result of our approach for the alternation-free μ-calculus coincides with the solution for the corresponding model checking problem.

Corollary 5.12 *Let $\mu Q.\phi$ be a closed μ-calculus formula in Alternation-free Normal Form. Assume that $\mu Q.\phi \longmapsto \langle cl_{\mu Q.\phi},\ pre_{\mu Q.\phi}\rangle$. For the least model ϱ of $cl_{\mu Q.\phi}$ such that $\varrho = \sqcap\{\varrho | (\varrho, \sigma)\ \underline{\textbf{sat}}\ cl_{\mu Q.\phi}, \varrho \supseteq \varrho_0\}$, where ϱ_0 defines P_p and T_a, we have $[\![\mu Q.\phi]\!] = \varrho(R_Q)$.*

Proof. It follows directly from Theorem 5.11. □

Example 5.5 *Consider a Kripke structure, given by the diagram to the left, where $S = \{s_1, s_2, s_3\}$, the transition relation $T = \{a\}$ is represented by edges labeled with a between states, and L labels s_1 with proposition p.*



$\varrho(R_Q)$	$[\![\mu Q.[a](p \vee Q)]\!]$
$\{s_1, s_3\}$	$\{s_1, s_3\}$

We evaluate the formula $\mu Q.[a](p \vee Q)$ over the above Kripke structure using ALFP and the semantics of the μ-calculus respectively. The results are given in the table to the right.

In our static analysis approach, we will first encode the above Kripke structure in ϱ_0 and then generate the clause $cl_{\mu Q.[a](p\vee Q)}$ for the formula $\mu Q.[a](p \vee Q)$ according to Table 5.1. We list this process as follows, where ALFP clauses of the form $\boldsymbol{true} \wedge cl$ has been simplified to cl. The least solution ϱ to $cl_{\mu Q.[a](p\vee Q)}$ subject to $\varrho_0 \subseteq \varrho$ can be calculated by succinct solver [29].

ϕ	cl_ϕ	pre_ϕ
p	$\boldsymbol{true}$	$P_p(s)$
Q	$\boldsymbol{true}$	$R_Q(s)$
$p \vee Q$	$\boldsymbol{true}$	$P_p(s) \vee R_Q(s)$
$[a](p \vee Q)$	$\boldsymbol{true}$	$\forall s' : \neg T_a(s, s') \vee P_p(s') \vee R_Q(s')$
$\mu Q.[a](p \vee Q)$	$\forall s : pre_{[a](p\vee Q)} \Rightarrow R_Q(s)$	$R_Q(s)$

5.3 Stratification Fails to Capture Syntactic Monotonicity

In this section, we analyze μ-calculus formulas of alternation depth 2 with the model checking approach and the approach we developed in Section 5.2 respectively. The main result of this section is that the solution to the model checking problem for μ-calculus formulas of alternation depth 2 cannot be characterised by a Moore Family result in ALFP.

To encode a closed μ-calculus formula ϕ into ALFP, we shall assume there must exist a clause defining the relation R_Q for each variable Q in ϕ. We focus on the rank of R_Q. We explain our negative result as follows in a more general way where we assign a rank to each variable Q in ϕ.

Given a formula ϕ of the μ-calculus and let the list of subformulas $\overrightarrow{\phi}$ be some ordering of all fixpoint subformulas of ϕ, i.e. $\overrightarrow{\mu Q.\mu R.(Q \vee R)} = (\mu Q.\mu R.(Q \vee R), \mu R.(Q \vee R))$. The model checking semantics of ϕ easily extends to $\overrightarrow{\phi}$, i.e. $[\![\overrightarrow{\mu Q.\mu R.(Q \vee R)}]\!] = ([\![\mu Q.\mu R.(Q \vee R)]\!], [\![\mu R.(Q \vee R)]\!]_{[Q\mapsto [\![\mu Q.\mu R.(Q\vee R)]\!]]})$.

Let ϕ be a closed formula of the μ-calculus. Assume that $\sigma Q_i.\phi_i$ (σ is either μ or ν) is a fixpoint subformula of ϕ $(1 \leq i \leq n)$. We define the function $F : \mathcal{P}(S)^n \to \mathcal{P}(S)^n$ by $F(S_1, ..., S_n) = ([\![\widetilde{\phi_1}]\!]_e, ..., [\![\widetilde{\phi_n}]\!]_e)$, where $e(Q_i) = S_i$, $\widetilde{\phi_i} = \phi_i[Q_j/\sigma Q_j.\phi_j]$ $(1 \leq j \leq n)$, and $\sigma Q_j.\phi_j$ is a top fixpoint subformula of ϕ_i. The notation $\phi_i[Q_j/\sigma Q_j.\phi_j]$ refers to a formula resulting from ϕ_i by substituting $\sigma Q_j.\phi_j$ with Q_j. We have the following theorem.

Theorem 5.13 *There exists a μ-calculus formula ϕ of alternation depth 2, where $Q_1, ..., Q_n$ is some ordering of all the variables in ϕ, such that $[\![\overrightarrow{\phi}]\!] = (S_1, ..., S_n)$ is not the least solution to the equation $F(S_1, ..., S_n) = (S_1, ..., S_n)$ with respect to $\sqsubseteq$ for any choice of ranking.*

Proof. Let $M = (S, T, L)$ be a Kripke structure , where $S = \{s_1, s_2\}$, $T = \{a\}$, $a = \{(s_1, s_2), (s_2, s_2)\}$, and L labels s_2 with proposition p. Consider the formula $\phi = \mu Q.(\neg \mu R.(R \vee (\neg Q \wedge p)))$. We can see that $ad(\phi) = 2$ once we translate ϕ to its Negation-free PNF.

We define $F(S_1, S_2) = ([\![\neg R]\!]_e, [\![R \vee (\neg Q \wedge p)]\!]_e)$, where $e(Q) = S_1$ and $e(R) = S_2$. Let's consider solutions to the equation $F(S_1, S_2) = (S_1, S_2)$. In the following, we use $\varrho(i)$ to denote the ith $(i = 1, 2)$ component in ϱ.

Let $\overrightarrow{\phi} = (\mu Q.(\neg \mu R.(R \vee (\neg Q \wedge p))), \mu R.(R \vee (\neg Q \wedge p)))$. According to the model checking semantics, we know that $\varrho_1 = [\![\overrightarrow{\phi}]\!] = ([\![\mu Q.(\neg \mu R.(R \vee (\neg Q \wedge p)))]\!], [\![\mu R.(R \vee (\neg Q \wedge p))]\!]_{e[Q \mapsto [\![\mu Q.(\neg \mu R.(R \vee (\neg Q \wedge p)))]\!]]}) = (\{s_1\}, \{s_2\})$. It's obvious that ϱ_1 is a solution to the equation$F(S_1, S_2) = (S_1, S_2)$. We also have another two solutions $\varrho_2 = (\emptyset, \{s_1, s_2\})$ and $\varrho_3 = (\{s_1, s_2\}, \emptyset)$ to it as well.

Since both $\varrho_2(1) \subset \varrho_1(1)$ and $\varrho_3(2) \subset \varrho_1(2)$ hold, it's obvious that ϱ_1 is not the least solution to the equation $F(S_1, S_2) = (S_1, S_2)$ with respect to $\sqsubseteq$ for any choice of ranking. $\square$

Theorem 5.13 can be extended to the case of a μ-calculus formula ϕ of alternation depth n $(n > 2)$. Whenever we develop a strongly benign translation to encode μ-calculus formulas to ALFP clauses, we implicitly define a function F above. Therefore, encoding the full μ-calculus formulas into ALFP using strongly benign translation is not feasible.

The negative result in the section suggests that we have to go beyond ALFP to characterize the full fragment of the μ-calculus. We continue this work in the next chapter and propose SFP which suffices to deal with the μ-calculus.

5.4 Future Work

In our future work, we are interesting in identifying fragments of the modal μ-calculus that reside properly between alternation depth 2 and alternation free for which the ALFP-based techniques might still work, i.e. for which the least fixed point can be described as a Moore family result in ALFP.

Chapter 6

The Modal μ-calculus in Succinct Fixed Point Logic

In Chapter 5, we have shown how to encode the model checking problem for the alternation-free μ-calculus in ALFP. However, as is suggested in the negative result there, ALFP is not well-suited for the encoding of the full fragment of the μ-calculus, where least and greatest fixed points are allowed to be mutually dependent on each other.

In this chapter, we continue the work of the previous chapter. We propose *Succinct Fixed Point Logic* (SFP) as an extension of ALFP and show that the model checking problem of the μ-calculus [2, 14] can be encoded in SFP. We first propose the notion of *weak stratification* which allows a convenient specification of nested fixed points in the μ-calculus. Then, we give the definition of the *intended model* of SFP clause sequences. We show through an example that we cannot take the greatest lower bound of the set of models of an SFP clause sequence as the intended model, since this does not match the fixed point semantics of the μ-calculus. Unlike in ALFP, we explicitly introduce a least fixed point operator in SFP to facilitate our development. Last, we explain our approach to the analysis of the μ-calculus and show that the intended model of an SFP clause sequence specifying a μ-calculus formula exactly characterizes the set of states which satisfy this μ-calculus formula over Kripke structures.

The structure of this chapter is as follows. We develop SFP in Section 6.1. Section 6.1.1 gives the framework of our logical approach to static analysis. This section mainly serves to provide a setting which makes the introduction of SFP more natural. When developing logic within this framework, we first need to consider a fragment of clause sequences and then establish an intended model for the fragment of clause sequences chosen. Section 6.1.2 gives the details of SFP. Section 6.2 shows the way to encode the model checking problem of the μ-calculus in SFP.

6.1 Succinct Fixed Point Logic

6.1.1 Logical Approach to Static Analysis

In our logical approach to static analysis, we specify analysis constraints in *clause sequences*. Assume that we are given a fixed countable set $\mathcal{X}$ of variables and a finite alphabet $\mathcal{R}$ of predicate symbols. We define the syntax of clause sequences cls, together with basic values v, pre-conditions pre and clauses cl as follows:

$$
\begin{array}{lcl}
v & ::= & c \mid x \\
pre & ::= & R(v_1, ..., v_n) \mid \neg R(v_1, ..., v_n) \mid pre_1 \wedge pre_2 \\
 & & \mid pre_1 \vee pre_2 \mid \forall x :\ pre \mid \exists x :\ pre \\
cl & ::= & R(v_1, ..., v_n) \mid \mathbf{true} \mid cl_1 \wedge cl_2 \mid pre \Rightarrow R(v_1, ..., v_n) \mid \forall x :\ cl \\
cls & ::= & cl_1, ..., cl_n
\end{array}
$$

The pre-conditions, clauses and clause sequences are interpreted over a finite and non-empty universe $\mathcal{U}$. A constant c is an element of $\mathcal{U}$, a variable $x \in \mathcal{X}$ ranges over $\mathcal{U}$, and the n-ary relation $R \in \mathcal{R}$ denotes a subset of $\mathcal{U}^n$. Occurrences of $R(v_1, ..., v_n)$ and $\neg R(v_1, ..., v_n)$ in pre-conditions are called *positive queries* and *negative queries*, respectively. All other occurrences of relations are *definitions* and often occur to the right of an implication.

Let $Int : \prod_k Rel_k \to \mathcal{P}(\mathcal{U}^k)$ be a mapping where Rel_k is a finite alphabet of k-ary predicate symbols and $\mathcal{P}(\mathcal{U}^k)$ is the powerset of $\mathcal{U}^k$. We define the satisfaction relations for pre-conditions, clauses and clause sequences

$$(\rho,\sigma)\ \underline{\texttt{sat}}\ \mathit{pre} \qquad \texttt{and} \qquad (\rho,\sigma)\ \underline{\texttt{sat}}\ \mathit{cl} \qquad \texttt{and} \qquad (\rho,\sigma)\ \underline{\texttt{sat}}\ \mathit{cls}$$

in Table 6.1, where $\rho \in \mathit{Int}$ is an interpretation of relations which maps each k-ary predicate symbol R to a subset of $\mathcal{U}^k$ and σ is an interpretation of variables. We write $\rho(R)$ for the set of k-tuples $(a_1, ...a_k)$ from $\mathcal{U}$ associated with the k-ary predicate R, we use $\sigma(x)$ to denote the atom of $\mathcal{U}$ bound to x and $\sigma[x \mapsto a]$ stands for the mapping that is σ except that x is mapped to a. We also treat a constant c as a variable by setting $\sigma(c) = c$.

$(\rho,\sigma)\ \underline{\texttt{sat}}\ R(v_1,...,v_n)$	`iff`	$(\sigma(v_1),...,\sigma(v_n)) \in \rho(R)$
$(\rho,\sigma)\ \underline{\texttt{sat}}\ \neg R(v_1,...,v_n)$	`iff`	$(\sigma(v_1),...,\sigma(v_n)) \notin \rho(R)$
$(\rho,\sigma)\ \underline{\texttt{sat}}\ \mathit{pre}_1 \wedge \mathit{pre}_2$	`iff`	$(\rho,\sigma)\ \underline{\texttt{sat}}\ \mathit{pre}_1$ `and` $(\rho,\sigma)\ \underline{\texttt{sat}}\ \mathit{pre}_2$
$(\rho,\sigma)\ \underline{\texttt{sat}}\ \mathit{pre}_1 \vee \mathit{pre}_2$	`iff`	$(\rho,\sigma)\ \underline{\texttt{sat}}\ \mathit{pre}_1$ `or` $(\rho,\sigma)\ \underline{\texttt{sat}}\ \mathit{pre}_2$
$(\rho,\sigma)\ \underline{\texttt{sat}}\ \forall x:\ \mathit{pre}$	`iff`	$(\rho,\sigma[x \mapsto a])\ \underline{\texttt{sat}}\ \mathit{pre}$ `for all` $a \in \mathcal{U}$
$(\rho,\sigma)\ \underline{\texttt{sat}}\ \exists x:\ \mathit{pre}$	`iff`	$(\rho,\sigma[x \mapsto a])\ \underline{\texttt{sat}}\ \mathit{pre}$ `for some` $a \in \mathcal{U}$
$(\rho,\sigma)\ \underline{\texttt{sat}}\ R(v_1,...,v_n)$	`iff`	$(\sigma(v_1),...,\sigma(v_n)) \in \rho(R)$
$(\rho,\sigma)\ \underline{\texttt{sat}}$ **true**	`iff`	**true**
$(\rho,\sigma)\ \underline{\texttt{sat}}\ \mathit{cl}_1 \wedge \mathit{cl}_2$	`iff`	$(\rho,\sigma)\ \underline{\texttt{sat}}\ \mathit{cl}_1$ `and` $(\rho,\sigma)\ \underline{\texttt{sat}}\ \mathit{cl}_2$
$(\rho,\sigma)\ \underline{\texttt{sat}}\ \mathit{pre} \Rightarrow R(v_1,...,v_n)$	`iff`	$(\rho,\sigma)\ \underline{\texttt{sat}}\ R(v_1,...,v_n)$ `whenever` $(\rho,\sigma)\ \underline{\texttt{sat}}\ \mathit{pre}$
$(\rho,\sigma)\ \underline{\texttt{sat}}\ \forall x:\ \mathit{cl}$	`iff`	$(\rho,\sigma[x \mapsto a])\ \underline{\texttt{sat}}\ \mathit{cl}$ `for all` $a \in \mathcal{U}$
$(\rho,\sigma)\ \underline{\texttt{sat}}\ \mathit{cl}_1,...,\mathit{cl}_n$	`iff`	$(\rho,\sigma)\ \underline{\texttt{sat}}\ \mathit{cl}_i$ `for all` i `where` $1 \le i \le n$

Table 6.1: Semantics of Pre-conditions, Clauses and Clause Sequences

A clause sequence with no free variables is called *closed*, and in closed clause sequences the interpretation σ is of no importance. For a fixed interpretation σ_0, when cls is closed, we have that $(\rho,\sigma)\ \underline{\texttt{sat}}\ \mathit{cls}$ agrees with $(\rho,\sigma_0)\ \underline{\texttt{sat}}\ \mathit{cls}$. We call an interpretation ρ a solution, or a model, of cls whenever $(\rho,\sigma_0)\ \underline{\texttt{sat}}\ \mathit{cls}$ holds.

Central to our approach to static analysis is the establishment of an *intended model* of cls. We often consider the least model of cls as a candidate, since that is the most precise analysis result. To deal with negations conveniently, we are often interested in some subsets of clause sequences defined by the above grammar. As can be seen from Chapter 2, ALFP actually restricts itself to the *stratified* fragment of clause sequences. The intended model of an ALFP formula is defined by the least model characterized by Moore Family properties. We propose *Succinct Fixed Point Logic* in the next section. SFP restricts itself to the *weakly stratified* fragment of clause sequences. The Moore Family result of SPF is established in a slightly different way and the model of an SFP formula is defined as the least model characterized by Moore Family properties as well.

6.1.2 Succinct Fixed Point Logic

The condition of stratification in ALFP requires that the definition of a relation R in cls only depends on relations with ranks less or equal to R. In particular, the requirement that a relation must be defined before they can be negatively queried is essential. This makes it inconvenient for ALFP to specify nested fixed points in the μ-calculus, where least and greatest fixed points are mutually dependent on each other.

In this section, we propose *Succinct Fixed Point Logic* (SFP) to encode nested fixed points in the μ-calculus. We first define the syntax of SFP, which include basic values v, pre-conditions pre, clauses cl, clause sequences cls and formulas f, as follows:

Definition 6.1 (Syntax of Succinct Fixed Point Logic)

$$\begin{array}{lcl} v & ::= & c \mid x \\ pre & ::= & R(v_1, ..., v_n) \mid \neg R(v_1, ..., v_n) \mid pre_1 \wedge pre_2 \\ & & \mid pre_1 \vee pre_2 \mid \forall x : \ pre \mid \exists x : \ pre \\ cl & ::= & R(v_1, ..., v_n) \mid \mathbf{true} \mid cl_1 \wedge cl_2 \mid pre \Rightarrow R(v_1, ..., v_n) \mid \forall x : \ cl \\ cls & ::= & cl_1, ..., cl_n \\ f & ::= & \mathbf{LFP}(cls) \end{array}$$

where cls is weakly stratified.

Here, we require that clause sequences are weakly stratified. The definition of *weak stratification* will be given later. We introduce a least fixed point operator $\mathbf{LFP}$ and $f = \mathbf{LFP}(cls)$ is defined as SFP formulas. This is mainly to facilitate the definition of the intended model of weakly stratified clause sequences. Our intention is that ρ is the intended model of cls iff ρ satisfies the formula $\mathbf{LFP}(cls)$.

To formalize the notion of weak stratification, we first give the definition of *Dependency Graph* as follows.

DEFINITION 6.2 (DEPENDENCY GRAPH) The dependency graph DG_{cls} of $cls = cl_1, ..., cl_n$ is a directed graph where each edge is labeled with a sign. The nodes of DG_{cls} are cl_1,...,cl_n. We define a positive (resp. negative) edge from cl_i to cl_j iff a relation defined in cl_i is positively (resp. negatively) queried in cl_j, where $1 \leq i, j \leq n$.

We say that cl_j *depends positively (resp. negatively)* on cl_i iff there exists a path in DG_{cls} from cl_i to cl_j with even (resp. odd) number of negative edges.

DEFINITION 6.3 (WEAK STRATIFICATION) A clause sequence $cls = cl_1$, ..., cl_n is *weakly stratified* iff the following conditions hold, where $1 \leq i, j \leq n$, $i \neq j$ and $R \in \mathcal{R}$:

- if R is defined in cl_i, then R is not defined in cl_j, and
- cl_i does not depend negatively on itself.
- if cl_i depends positively (resp. negatively) on cl_j, then cl_i does not depend negatively (resp. positively) on cl_j.

The first condition in the above definition simply says that we use only one clause to define each relation. The second condition imposes *syntactic monotonicity* to the clause sequence. The last condition is actually used to facilitate the establishment of a Moore Family result for SFP.

EXAMPLE 6.1 *The following clause sequence satisfies the condition of weak stratification.*

$$cls = (\forall x : \neg R_2(x) \Rightarrow R_1(x)), (\forall x : \neg R_1(x) \Rightarrow R_2(x))$$

EXAMPLE 6.2 *The following clause sequence is ruled out by the notion of weak stratification. We can see that the clause* $(\forall x : R_2(x) \Rightarrow R_1(x))$ *depends negatively on itself.*

$$cls = (\forall x : R_2(x) \Rightarrow R_1(x)), (\forall x : \neg R_1(x) \Rightarrow R_2(x))$$

Let's consider the following example where we specify a μ-calculus formula of nested fixed points with a weakly stratified clause sequence.

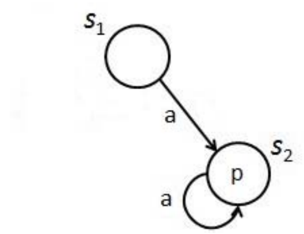
EXAMPLE 6.3 *Consider the μ-calculus formula $\phi = \mu Q_1.(\neg\mu Q_2.(Q_2 \vee (\neg Q_1 \wedge p)))$, which is semantically equivalent to $\mu Q_1.(\nu Q_2.(Q_2 \wedge (Q_1 \vee \neg p)))$ and therefore consists of nested fixed points. This is actually an alternation depth 2 formula. We can see that the least fixed point μQ_1 and the greatest fixed point νQ_2 are mutually dependent on each other. The formula ϕ can be specified by the following clause sequence cls.*

$$cls = [\forall s : \neg R_{Q_2}(s) \Rightarrow R_{Q_1}(s)], [\forall s : [R_{Q_2}(s) \vee (\neg R_{Q_1}(s) \wedge P_p(s))] \Rightarrow R_{Q_2}(s)]$$

The clause sequence cls is weakly stratified. The relation P_p intends to specify the set of states, in a given Kripke structure, on which the atomic proposition p holds. The relation R_{Q_1} (resp. R_{Q_2}) intends to characterize $[\![\phi]\!]_{[]}$ (resp. $[\![\mu Q_2.(Q_2 \vee (\neg Q_1 \wedge p))]\!]_{[Q_1 \mapsto [\![\phi]\!]_{[]}]}$).

The next step is to define an intended model ρ of cls. In our setting, this amounts to define the semantics of formulas $f = \mathbf{LFP}(cls)$. Our intention is to use ρ to encode the fixed point semantics in the μ-calculus. Our first try is to define it in a similar way as we do in ALFP. Let's assume that all relations defined in a clause cl_i have the same rank and that all predefined relations have rank 0. However, we show through the following example that we cannot define the intended model ρ of cls as $\sqcap\{\rho|(\rho, \sigma_0)\ \underline{\texttt{sat}}\ cls \wedge \rho_0 \subseteq \rho\}$, where ρ_0 defines all predefined relations, with respect to $\sqsubseteq$, since it does not capture the fixed point semantics.

EXAMPLE 6.4 *Consider the Kripke structure $M = (S, T, L)$, given by the diagram to the left, where $S = \{s_1, s_2\}$, $T = \{a\}$, $a = \{(s_1, s_2), (s_2, s_2)\}$, and L labels s_2 with the proposition p. We encode the μ-calculus formula $\phi = \mu Q_1.(\neg\mu Q_2.(Q_2 \vee (\neg Q_1 \wedge p)))$ in the same clause sequence $cls = [\forall s : \neg R_{Q_2}(s) \Rightarrow R_{Q_1}(s)], [\forall s : [R_{Q_2}(s) \vee (\neg R_{Q_1}(s) \wedge P_p(s))] \Rightarrow R_{Q_2}(s)]$ as we do in Example 6.3. We evaluate ϕ over M using SFP and the semantics of the μ-calculus respectively.*



	ρ_1	ρ_2	ρ_3
R_{Q_2}	$\{s_1, s_2\}$	$\emptyset$	$\{s_2\}$
R_{Q_1}	$\emptyset$	$\{s_1, s_2\}$	$\{s_1\}$
P_p	$\{s_2\}$	$\{s_2\}$	$\{s_2\}$

Assume we have an initial interpretation ρ_0, where $\rho_0(P_p) = \{s_2\}$ and $\rho_0(R_{Q_1}) = \rho_0(R_{Q_2}) = \emptyset$. We now consider the set of interpretations $I = \{\rho | (\rho, \sigma_0)$ $\underline{\textbf{\textit{sat}}}$ $cls \wedge \rho_0 \subseteq \rho\}$ according to the semantics in Table 6.1. There are at least three solutions ρ_1, ρ_2 and ρ_3, given in the table to the right, in the set I.

We can take at most three essentially different ranking functions $rank_1$, $rank_2$ and $rank_3$. The function $rank_1$ is defined by $rank_1(P_p) = 0$, $rank_1(R_{Q_1}) = 1$ and $rank_1(R_{Q_2}) = 2$. The function $rank_2$ is defined by $rank_2(P_p) = 0$, $rank_2(R_{Q_1}) = 2$ and $rank_2(R_{Q_2}) = 1$. The function $rank_3$ is defined by $rank_3(P_p) = 0$, $rank_3(R_{Q_1}) = 1$ and $rank_3(R_{Q_2}) = 1$.

Let $e = [Q_1 \mapsto [\![\phi]\!]_{[]}, Q_2 \mapsto [\![\mu Q_2.(Q_2 \vee (\neg Q_1 \wedge p))]\!]_{[Q_1 \mapsto [\![\phi]\!]_{[]}]}]$. According to the semantics of the μ-calculus, we know that $[\![Q_1]\!]_e = \{s_1\}$ and $[\![Q_2]\!]_e = \{s_2\}$. We can see that ρ_3 exactly characterizes the semantics of the μ-calculus in our example. However, due to the existence of ρ_1 and ρ_2, the solution ρ_3 is not the least model in I for either $rank_1$ or $rank_2$ or $rank_3$.

The method of establishing an intended model of cls in the above example can be summarized as follows. First, we calculate all the models that satisfy cls. Second, we make a choice of ranks for all those relations defined in cls. Last, we choose the least model as the intended model of cls, according to the lexicographic ordering with respect to the choice of ranks we have made. This method applies well when we approximate an analysis where analysis information only flows from the lowest rank to the highest rank. Therefore, ALFP successfully characterizes the semantics of the alternation-free μ-calculus (see the previous chapter for details).

In the following, we define the semantics of formulas f. We assume that $cls = cl_1, ..., cl_n$ and write $\rho = \varrho_0, \varrho_1, ..., \varrho_n$ to mean that ϱ_0 is an interpretation for some predefined relations and ϱ_i ($1 \leq i \leq n$) is an interpretation of relations defined in cl_i. We use $\rho[\varrho'_i/\varrho_i]$ to denote a new interpretation updated from ρ by substituting ϱ_i with ϱ'_i. Let ϱ_i and ϱ'_i be two interpretations of relations defined in cl_i. We define that $\varrho_i \subseteq \varrho'_i$ iff for all relations R defined in cl_i, $\varrho_i(R) \subseteq \varrho'_i(R)$ holds. The set of interpretations of relations defined in cl_i constitute a complete lattice with respect to $\subseteq$. The satisfaction relation (ρ, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_1, ..., cl_n)$ is defined in the following.

Definition 6.4 (Semantics of SFP formulas) Let $\rho = \varrho_0, ..., \varrho_n$ be an interpretation and $cls = cl_1, ..., cl_n$ a weakly stratified clause sequence. The satisfaction relation (ρ, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_1, ..., cl_n)$ is defined inductively as follows:

- (ρ, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_n)$ iff $\varrho_n = \sqcap\{\varrho'_n \mid (\rho[\varrho'_n/\varrho_n], \sigma)$ $\underline{\texttt{sat}}$ $cl_n\}$
- (ρ, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_i, ..., cl_n)$ iff
  1. (ρ, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_{i+1}, ..., cl_n)$, and
  2. $\varrho_i = \sqcap\{\varrho'_i \mid \exists \varrho'_{i+1}, ..., \varrho'_n : (\rho[\varrho'_i/\varrho_i, ..., \varrho'_n/\varrho_n], \sigma)$ $\underline{\texttt{sat}}$ $cl_i \wedge$ $(\rho[\varrho'_i/\varrho_i, ..., \varrho'_n/\varrho_n], \sigma)$ $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_{i+1}, ..., cl_n)\}$

The Moore Family properties for weakly stratified clause sequence $cls = cl_1, ..., cl_n$ is established as follows.

Theorem 6.5 *Let $\rho = \varrho_0, ..., \varrho_n$ be an interpretation, $cls = cl_1, ..., cl_n$ a weakly stratified clause sequence and $1 \leq i \leq n$. Then, we have the followings:*

- *The set of interpretations $\{\varrho'_n \mid (\rho[\varrho'_n/\varrho_n], \sigma)$ $\underline{\textit{sat}}$ $cl_n\}$ is a Moore Family*
- *The set of interpretations $\{\varrho'_i \mid \exists \varrho'_{i+1}, ..., \varrho'_n : (\rho[\varrho'_i/\varrho_i, ..., \varrho'_n/\varrho_n], \sigma)$ $\underline{\textit{sat}}$ $cl_i \wedge$*

  $(\rho[\varrho'_i/\varrho_i, ..., \varrho'_n/\varrho_n], \sigma)$ $\underline{\textit{sat}}$ $\boldsymbol{LFP}(cl_{i+1}, ..., cl_n)\}$ is a Moore Family.

Proof. In Appendix D. □

We define the intended model of a weakly stratified clause sequence below.

Definition 6.6 Assume that $cls = cl_1, ..., cl_n$ is a weakly stratified clause sequence. The model ρ is an intended model of cls iff (ρ, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_1, ..., cl_n)$.

The Moore Family properties of SFP leads to the following theorem which guarantees the existence and the uniqueness of the intended model of cls.

Theorem 6.7 *Let $cls = cl_1, ..., cl_n$ be a weakly stratified clause sequence. The model ρ such that (ρ, σ) $\underline{\texttt{sat}}$ $\boldsymbol{LFP}(cl_1, ..., cl_n)$ exists and is unique.*

Proof. Based on Theorem 6.5, the proof is a simple induction on the number of clauses in $cls = cl_1, ..., cl_n$. □

Example 6.5 *Let's reconsider the problem in Example 6.4 again and show how to find the model $\rho = \varrho_0, \varrho_1, \varrho_2$ to the formula $\boldsymbol{LFP}(cls)$. Let's write $cls = cl_1, cl_2$ where $cl_1 = [\forall s : \neg R_{Q_2}(s) \Rightarrow R_{Q_1}(s)]$ and $cl_2 = [\forall s : [R_{Q_2}(s) \vee (\neg R_{Q_1}(s) \wedge P_p(s))] \Rightarrow R_{Q_2}(s)]$. From Definition 6.4, (ρ, σ) $\underline{\texttt{sat}}\boldsymbol{LFP}(cl_1, cl_2)$ iff (ρ, σ) $\underline{\texttt{sat}}$ $\boldsymbol{LFP}(cl_2)$ and $\varrho_1 = \sqcap\{\varrho_1' \mid \exists \varrho_2' : (\rho[\varrho_1'/\varrho_1, \varrho_2'/\varrho_2], \sigma)$ $\underline{\texttt{sat}}$ $cl_1 \wedge (\rho[\varrho_1'/\varrho_1, \varrho_2'/\varrho_2], \sigma)$ $\underline{\texttt{sat}}$ $\boldsymbol{LFP}(cl_2)\}$.*

We first calculate the set of interpretations such that (ρ, σ) $\underline{\texttt{sat}}$ $\boldsymbol{LFP}(cl_2)$. To this end, we first list all the interpretations such that (ρ, σ) $\underline{\texttt{sat}}$ cl_2 in Table 6.2. In this case, relations P_p and R_{Q_1} are predefined relations for the clause cl_2.

	ρ_1	ρ_2	ρ_3	ρ_4	ρ_5	ρ_6	ρ_7	ρ_8	ρ_9
R_{Q_2}	$\{s_2\}$	$\{s_1, s_2\}$	$\{s_2\}$	$\{s_1, s_2\}$	$\emptyset$	$\{s_1$	$\{s_2\}$	$\{s_1, s_2\}$	$\emptyset$
R_{Q_1}	$\emptyset$	$\emptyset$	$\{s_1\}$	$\{s_1\}$	$\{s_2\}$	$\{s_2\}$	$\{s_2\}$	$\{s_2\}$	$\{s_1, s_2\}$
P_p	$\{s_2\}$	$\{s_2\}$	$\{s_2\}$	$\{s_2\}$	$\{s_2\}$	$\{s_2\}$	$\{s_2\}$	$\{s_2\}$	$\{s_2\}$

ρ_{10}	ρ_{11}	ρ_{12}
$\{s_1\}$	$\{s_2\}$	$\{s_1, s_2\}$
$\{s_1, s_2\}$	$\{s_1, s_2\}$	$\{s_1, s_2\}$
$\{s_2\}$	$\{s_2\}$	$\{s_2\}$

Table 6.2: (ρ, σ) $\underline{\texttt{sat}}$ cl_2

The next step is to select those interpretations which satisfy $\boldsymbol{LFP}(cl_2)$ from Table 6.2. From all those interpretations which coincide on predefined relations, we choose the one with the best analysis result for R_{Q_2}. Let's take ρ_1 and ρ_2 as an example. The models ρ_1 and ρ_2 coincide on their interpretations for P_p and R_{Q_1}. However, $\rho_1(R_{Q_2}) = \sqcap\{\rho_1(R_{Q_2}), \rho_2(R_{Q_2})\}$. Therefore, (ρ_1, σ) $\underline{\texttt{sat}}$ $\boldsymbol{LFP}(cl_2)$. The result of our selection are $\{\rho_1, \rho_3, \rho_5, \rho_9\}$. These are the interpretations *which satisfy* $\boldsymbol{LFP}(cl_2)$.

We now select those interpretations which satisfy cl_1 *from* $\{\rho_1, \rho_3, \rho_5, \rho_9\}$ *and see that only* ρ_3 *and* ρ_9 *do. The last step is to select from* ρ_3 *and* ρ_9 *the one which satisfies* $\boldsymbol{LFP}(cl_1, cl_2)$. *Since* $\rho_3(R_{Q_1}) = \sqcap\{\rho_3(R_{Q_1}), \rho_9(R_{Q_1})\}$, *we know that* (ρ_3, σ) $\underline{\texttt{sat}}$ $\boldsymbol{LFP}(cl_1, cl_2)$. *Notice that* ρ_3 *exactly characterized the fixed point semantics here.*

6.2 Modal μ-calculus in SFP

We first give another syntax of the μ-calculus using only the μ operator as follows, which will facilitate our static analysis approach to the analysis of the μ-calculus.

Definition 6.8 Let Var be a set of variables, $\mathbf{P}$ be a set of atomic propositions that is closed under negation. The syntax of the μ-calculus is defined as follows:

$$\phi ::= \; p \mid Q \mid \neg Q \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid \langle a \rangle \phi \mid [a]\phi \mid \mu Q.\phi \mid \neg \mu Q.\phi$$

where no variable is quantified twice and ϕ is syntactically monotone in Q in the cases of $\mu Q.\phi$ and $\neg \mu Q.\phi$.

Here, we use Definition 6.8 to give the syntax of the μ-calculus. Given a μ-calculus formula ϕ, for each variable Q in ϕ, a relation R_Q is defined. We specify our analysis with a pair $\langle cls_\phi, pre_\phi \rangle$, where cls_ϕ is a weakly stratified clause sequence and pre_ϕ is a pre-condition.

Assume that $\rho = \varrho_0, ..., \varrho_n$ such that (ρ, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_\phi)$, where ϱ_0 is an initial interpretation which encodes a given Kripke structure and defines relations $R_{Q_1}, ..., R_{Q_n}$, where $Q_1, ..., Q_n$ are all the free variables in ϕ. The intention of our development is that $s' \in [\![\phi]\!]_{e[Q_1 \mapsto S_1, ..., Q_n \mapsto S_n]}$ iff $(\rho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_ϕ, and that when ϕ takes the form $\mu Q.\phi$, we have that $[\![\mu Q.\phi]\!]_{e[Q_1 \mapsto S_1, ..., Q_n \mapsto S_n]}$ equals $\rho(R_Q)$.

We encode a Kripke structure $M = (S, T, L)$ into SFP by defining the corresponding relations in ϱ_0 as follows. Assume that the universe is $\mathcal{U} = S$,

- for each atomic proposition p we define a predicate P_p such that $s \in \varrho_0(P_p)$ if and only if $p \in L(s)$,
- for each element a in T, we define a binary relation T_a such that $(s,t) \in \varrho_0(T_a)$ if and only if $(s,t) \in a$.

The mapping rules for $\phi \longmapsto \langle cls_\phi,\ pre_\phi \rangle$ is given in Table 6.3. The clause sequence cls_ϕ is used to define all the relations R_Q where Q is a bounded variable in ϕ. We use $pre_\phi[s'/s]$ to denote a pre-condition resulting from pre_ϕ by substituting the free variable s in pre_ϕ with s'.

In Table 6.3, the choice of the ordering of clauses in cls_ϕ is essential in our approach. Assume that $cls_\phi = cl_1, ..., cl_n$. We define only one relation in each clause cl_i $(1 \leq i \leq n)$. Assume that we are given a μ-calculus formula ϕ. We call a subformula of ϕ a μ-subformula iff its main connective is μ. Assume that $\mu Q_i.\varphi_1$ and $\mu Q_j.\varphi_2$ are two μ-subformulas in ϕ and we define R_{Q_i} (resp. R_{Q_j}) in cl_i (resp. cl_j), our intention is to ensure that $i < j$ if $\mu Q_j.\varphi_2$ is a proper subformula of $\mu Q_i.\varphi_1$. Therefore, in the case of $\mu Q.\phi \longmapsto \langle cls_\phi,\ pre_\phi \rangle$, for example, we have that $cls_{\mu Q.\phi} = ([\forall s : pre_\phi \Rightarrow R_Q(s)], cls_\phi)$ instead of $cls_{\mu Q.\phi} = (cls_\phi, [\forall s : pre_\phi \Rightarrow R_Q(s)])$.

We first explain the case of $\mu Q.\phi$. Here, Q is a bounded variable. Under the assumption that $\phi \longmapsto \langle cls_\phi,\ pre_\phi \rangle$ holds, we define $cls_{\mu Q.\phi}$ as $([\forall s : pre_\phi \Rightarrow R_Q(s)], cls_\phi)$. The clause $[\forall s : pre_\phi \Rightarrow R_Q(s)]$ defines the relation R_Q and the clause sequence cls_ϕ defines all those relations $R_{Q'}s$ where Q' is a bounded variable in ϕ. We define $pre_{\mu Q.\phi}$ as $R_Q(s)$.

For atomic proposition p, we simply define cls_p as **true** since there are no bounded variables in p. We make use of the predefined predicate P_p and define pre_p as $P_p(s)$. For a variable Q, we also define cls_Q as **true** since the Q is a free variable here. We define pre_Q as $R_Q(s)$. For $\neg Q$, we define $cls_{\neg Q}$ as **true** and define $pre_{\neg Q}$ as $\neg R_Q(s)$.

For $\phi_1 \vee \phi_2$, we assume that $\phi_1 \longmapsto \langle cls_{\phi_1},\ pre_{\phi_1} \rangle$ and $\phi_2 \longmapsto \langle cls_{\phi_2},\ pre_{\phi_2} \rangle$. This means that for each subformula $\mu Q.\phi$ in ϕ_1 (resp. ϕ_2), the relation R_Q is

$$
\begin{array}{lll}
p & \longmapsto & \langle \mathbf{true},\ P_p(s)\rangle \\
Q & \longmapsto & \langle \mathbf{true},\ R_Q(s)\rangle \\
\neg Q & \longmapsto & \langle \mathbf{true},\ \neg R_Q(s)\rangle \\
\phi_1 \vee \phi_2 & \longmapsto & \langle (cls_{\phi_1}, cls_{\phi_2}),\ pre_{\phi_1} \vee pre_{\phi_2}\rangle \\
 & & \texttt{whenever } \phi_1 \longmapsto \langle cls_{\phi_1},\ pre_{\phi_1}\rangle \texttt{ and } \phi_2 \longmapsto \langle cls_{\phi_2},\ pre_{\phi_2}\rangle \\
\phi_1 \wedge \phi_2 & \longmapsto & \langle (cls_{\phi_1}, cls_{\phi_2}),\ pre_{\phi_1} \wedge pre_{\phi_2}\rangle \\
 & & \texttt{whenever } \phi_1 \longmapsto \langle cls_{\phi_1},\ pre_{\phi_1}\rangle \texttt{ and } \phi_2 \longmapsto \langle cls_{\phi_2},\ pre_{\phi_2}\rangle \\
\langle a\rangle\phi & \longmapsto & \langle cls_{\phi},\ \exists s' : T_a(s,s') \wedge pre_{\phi}[s'/s]\rangle \\
 & & \texttt{whenever } \phi \longmapsto \langle cls_{\phi},\ pre_{\phi}\rangle \\
[a]\phi & \longmapsto & \langle cls_{\phi},\ \forall s' : \neg T_a(s,s') \vee pre_{\phi}[s'/s]\rangle \\
 & & \texttt{whenever } \phi \longmapsto \langle cls_{\phi},\ pre_{\phi}\rangle \\
\mu Q.\phi & \longmapsto & \langle ([\forall s : pre_{\phi} \Rightarrow R_Q(s)], cls_{\phi}),\ R_Q(s)\rangle \\
 & & \texttt{whenever } \phi \longmapsto \langle cls_{\phi},\ pre_{\phi}\rangle \\
\neg\mu Q.\phi & \longmapsto & \langle cls_{\mu Q.\phi},\ \neg R_Q(s)\rangle \\
 & & \texttt{whenever } \mu Q.\phi \longmapsto \langle cls_{\mu Q.\phi},\ pre_{\mu Q.\phi}\rangle
\end{array}
$$

Table 6.3: μ-calculus in Succinct Fixed Point Logic

defined in cls_{ϕ_1} (resp. cls_{ϕ_2}) and that pre_{ϕ_1} and pre_{ϕ_2} are also defined as expected. We define $cls_{\phi_1 \vee \phi_2}$ as $(cls_{\phi_1}, cls_{\phi_2})$. This ensures that for each bounded variable Q in $\phi_1 \vee \phi_2$, R_Q is defined in $(cls_{\phi_1}, cls_{\phi_2})$. It's natural to define $pre_{\phi_1 \vee \phi_2}$ as $pre_{\phi_1} \vee pre_{\phi_2}$. The case for $\phi_1 \wedge \phi_2$ follows the same pattern.

For $\langle a\rangle\phi$, we assume that $\phi \longmapsto \langle cls_{\phi},\ pre_{\phi}\rangle$. We simply define that $cls_{\langle a\rangle\phi} = cls_{\phi}$ and this suffices to guarantee that for each bounded variable Q in $\langle a\rangle\phi$, the relation R_Q is defined in $cls_{\langle a\rangle\phi}$. We define $pre_{\langle a\rangle\phi}$ as $\exists s' : T_a(s,s') \wedge pre_{\phi}[s'/s]$. This means for any state s if $pre_{\phi}[s'/s]$ holds on any of the a-derivative s' of s, then $pre_{\langle a\rangle\phi}$ holds on state s. This matches the semantics for $\langle a\rangle\phi$.

For $[a]\phi$, we also assume that $\phi \longmapsto \langle cls_{\phi},\ pre_{\phi}\rangle$. For a similar reason as in the case for $\langle a\rangle\phi$, we define that $cls_{[a]\phi} = cls_{\phi}$. We define $pre_{[a]\phi}$ by $\forall s' : \neg T_a(s,s') \vee pre_{\phi}[s'/s]$. This means for any state s if $pre_{\phi}[s'/s]$ holds on all of the a-derivatives s' of s, then $pre_{[a]\phi}$ holds on state s.

For $\neg\mu Q.\phi$, we assume that $\mu Q.\phi \longmapsto \langle cls_{\mu Q.\phi},\ pre_{\mu Q.\phi}\rangle$. We define that $cls_{\neg\mu Q.\phi} = cls_{\mu Q.\phi}$. We simply define $pre_{\neg\mu Q.\phi}$ as $\neg R_Q(s)$.

We have the following lemma which ensures that our specification of the μ-calculus formulas is within SFP.

Lemma 6.9 *Given a closed μ-calculus formula ϕ, assume that $\phi \longmapsto \langle cls_\phi,\ pre_\phi \rangle$ holds according to Table 6.3, the clause sequence cls_ϕ is closed and weakly stratified.*

Proof. In Appendix D. $\square$

The following theorem shows that the pre-condition pre_ϕ in our mapping $\phi \longmapsto \langle cls_\phi,\ pre_\phi \rangle$ correctly characterizes the semantics of ϕ.

Theorem 6.10 *Let ϕ be a μ-calculus formula with $Q_1, ..., Q_n$ being all the free variables in it. Assume that $\phi \longmapsto \langle cls_\phi,\ pre_\phi \rangle$. Let $\rho = \varrho_0, ..., \varrho_n$ be an interpretation such that (ρ, σ) $\underline{\textbf{sat}}$ $\textbf{LFP}(cls_\phi)$, where $\varrho_0(R_{Q_1}) = S_1, ..., \varrho_0(R_{Q_n}) = S_n$ and ϱ_0 defines P_p and T_a. Then, $s' \in [\![\phi]\!]_{e[Q_1 \mapsto S_1, ..., Q_n \mapsto S_n]}$ iff $(\rho, \sigma[s \mapsto s'])$ $\underline{\textbf{sat}}$ pre_ϕ.*

Proof. In Appendix D. $\square$

We focus on closed μ-calculus formulas of the form $\mu Q.\phi$. As is mentioned in the previous chapter, this is not a restriction since $[\![\phi]\!] = [\![\mu Q.\phi]\!]$ when Q is not a free variable in ϕ. From Theorem 6.10, we have the following corollaries saying that the model of SFP formulas for the analysis of the μ-calculus coincides with the solution for the corresponding model checking problem.

Corollary 6.11 *Let $\mu Q.\phi$ be a closed μ-calculus formula. Assume that $\mu Q.\phi \longmapsto \langle cl_{\mu Q.\phi},\ pre_{\mu Q.\phi} \rangle$ holds. Let $\rho = \varrho_0, ..., \varrho_n$ be an interpretation such that (ρ, σ) $\underline{\textbf{sat}}$ $\textbf{LFP}(cls_{\mu Q.\phi})$, where ϱ_0 defines P_p and T_a. Then, we have that $[\![\mu Q.\phi]\!] = \rho(R_Q)$.*

Proof. It follows directly from Theorem 6.10. $\square$

6.3 Future Work

We have proposed Succinct Fixed Point Logic in this chapter. In our future work, we are interested in developing efficient solvers for SFP so that model checkers for the μ-calculus are implicitly implemented as well.

Chapter 7

Conclusion

In this thesis, we have developed several static analysis techniques to deal with model checking problems.

A number of papers (surveyed in [15, 31]) have developed flow logic as a uniform approach to static analysis using in particular 2-valued Alternation-free Least Fixed Point Logic (ALFP) as the specification language; on top of the many theoretical results established for this approach also a number of solvers have been developed [51] that make it easy to obtain prototype implementations.

Based on the work of [21], we first developed a flow logic approach to static analysis and encoded CTL model checking without fairness assumptions into 2-valued ALFP. Then, we start to deal with the fairness assumptions in CTL model checking. It is shown in [10] that computing nontrivial strongly connected components of transition systems plays an important role in solving different types of fairness constraints in CTL. ALFP is a type of least fixed point logic which is able to calculate the transitive closure of a relation. Our ideas of calculating nontrivial strongly connected sets are mainly based on calculating transitive closure of the transition relations of Kripke structures. Our encoding works well for unconditional and weak fairness constraints. However, an exponential blow up occurs when we deal with strong fairness constraints. Therefore, our ALFP-based is not well suited for solving a large number of strong fairness constraints.

In two-valued setting, we also considered the model checking problem for the alternation-free μ-calculus, which is more expressive than CTL [6]. Our positive result there has shown that ALFP suffices to encode the model checking problem for the alternation-free μ-calculus. However, our negative result is that the full μ-calculus cannot be encoded in a similar way regardless of the choice of ranking. It would be interesting to identify fragments of the modal μ-calculus that reside properly between alternation depth 2 and alternation free for which the ALFP-based development might still work, i.e. for which the least fixed point can be described as a Moore family result in ALFP.

In many approaches to analyzing systems it has been realized that it is useful to consider multi-valued logics; in model checking this includes the developments of [40, 49, 56, 41, 64, 42, 43, 44, 45] and in static analysis a notable contribution is [54, 55, 57, 46, 47]. To generalize our ALFP-based static analysis approach to a multi-valued setting, we proposed multi-valued ALFP. We established a Moore family result ensuring the existence of best solutions even in the case of negation as long as a notion of stratification is adhered to. We also showed that existing solvers can be used also for the generalized case since multi-valued ALFP, when interpreted over a finite distributive multi-valued structure, can be "translated" into 2-valued ALFP. Finally, we showed that our approach can be used to analyze Computation Tree Logic (CTL) in the multi-valued setting thereby generalizing the work in Chapter 3 and [21]. In our future work, we are interested in comparing our multi-valued analysis result with the semantics of multi-valued CTL proposed in [43]. We are also motivated in introducing fairness assumptions into the multi-valued setting and developing a multi-valued analysis for CTL with different fairness constraints.

To encode the full fragment of the μ-calculus, we proposed SFP as an extension of ALFP. ALFP can be encoded in SFP by showing that the least model of an ALFP formula can be characterized as the model of a corresponding SFP formula. This encoding is conceptually obvious and is not given in this thesis. We showed that μ-calculus formulas of nested fixed points can be characterized as the intended model of SFP clause sequences. Currently, SFP is not supported by any solvers. In our future work, we are interested in developing an efficient solver to calculate the model for SFP formulas so that a model checker for the μ-calculus is also implicitly implemented.

Appendix A

Appendix for Chapter 3

Lemma 3.1 The ALFP clauses generated for judgements $\vec{R} \vdash \phi$ defined in Table 3.1 are closed and stratified.

Proof. It is easy to see that clauses $\vec{R} \vdash \phi$ defined in Table 3.1 are indeed closed for any CTL formula ϕ. It is also straightforward that the ALFP clauses for $\vec{R} \vdash \phi$ only contain definitions of relations of ranks in $\{0, ..., depth(\phi)\}$ and it only use relations of ranks in $\{0, ..., depth(\phi)\}$. All negative uses of relations in ALFP clauses generated for judgement $\vec{R} \vdash \neg\varphi$ only involve relations of ranks $\{0, ..., depth(\neg\varphi) - 1\}$. This gives us the intuition to proceed our proof. To prove that the ALFP clauses generated for the judgement $\vec{R} \vdash \phi$ are stratified, we prove by structural induction on ϕ.

Base cases:

Case $\phi = true$: The clause we get is $\forall s : R_{true}(s)$. Since $rank_{R_{true}} = depth(true) = 0$, it's obvious that stratification is guaranteed.

Case $\phi = p$: The clause we get is $\forall s : P_p(s) \Rightarrow R_p(s)$. Since $rank_{P_p} = 0$ and

$rank_{R_p} = depth(p) = 0$, stratification is also guaranteed.

For boolean connectives:

Case $\phi = \neg\phi'$: The clause we get consists of two parts, namely the clause generated for $\vec{R} \vdash \phi'$ and $\forall s : (\neg R_{\phi'}(s)) \Rightarrow R_{\neg\phi'}(s)$. According to the induction hypothesis, the clause generated for $\vec{R} \vdash \phi'$ is stratified. Since $rank_{R_{\neg\phi'}} = depth(\neg\phi') = 1 + depth(\phi') = 1 + rank_{R_{\phi'}} > rank_{R_{\phi'}}$, therefore according to the definition of stratification, we know that the clause for $\vec{R} \vdash \neg\phi'$ is also stratified.

Case $\phi = \phi_1 \vee \phi_2$: The clause we get also consists of two parts, namely the clause generated for $\vec{R} \vdash \phi_1 \wedge \vec{R} \vdash \phi_2$ and $\forall s : R_{\phi_1}(s) \vee R_{\phi_2}(s) \Rightarrow R_{\phi_1 \vee \phi_2}(s)$. According to the induction hypothesis, the clause for $\vec{R} \vdash \phi_1 \wedge \vec{R} \vdash \phi_2$ is stratified. Since $rank_{R_{\phi_1 \vee \phi_2}} = depth(\phi_1 \vee \phi_2) = 1 + max\{depth(\phi_1), depth(\phi_2)\} = 1 + max\{rank_{R_{\phi_1}}, rank_{R_{\phi_2}}\}$, we have $rank_{R_{\phi_1 \vee \phi_2}} > rank_{R_{\phi_1}}$ and $rank_{R_{\phi_1 \vee \phi_2}} > rank_{R_{\phi_2}}$. According to the definition of stratification, we know that the clause for $\vec{R} \vdash \phi_1 \vee \phi_2$ is also stratified.

For modal operators:

Case $\phi = \mathbf{EX}\phi'$: The clause we get consists of two parts, namely the clause generated for $\vec{R} \vdash \phi'$ and $\forall s : [\exists s' : T(s, s') \wedge R_{\phi'}(s')] \Rightarrow R_{\mathbf{EX}\phi'}(s)$. According to the induction hypothesis, the clause generated for $\vec{R} \vdash \phi'$ is stratified. Since $rank_{R_{\mathbf{EX}\phi'}} = depth(\mathbf{EX}\phi') = 1 + depth(\phi') = 1 + rank_{R_{\phi'}} > rank_{R_{\phi'}}$ and $rank_T = 0$, therefore according to the definition of stratification, we know that the clause for $\vec{R} \vdash \mathbf{EX}\phi'$ is stratified.

Case $\phi = \mathbf{E}[\phi_1 \mathbf{U} \phi_2]$: The clause we get consists of two parts, namely the clause generated for $\vec{R} \vdash \phi_1 \wedge \vec{R} \vdash \phi_2$ and $[\forall s : R_{\phi_2}(s) \Rightarrow R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}(s)] \wedge [\forall s : [\exists s' : T(s, s') \wedge R_{\phi_1}(s) \wedge R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}(s')] \Rightarrow R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}(s)]$. According to the induction hypothesis, the clause for $\vec{R} \vdash \phi_1 \wedge \vec{R} \vdash \phi_2$ is stratified. Since $rank_{R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}} = depth(\mathbf{E}[\phi_1 \mathbf{U} \phi_2]) = 1 + max\{depth(\phi_1), depth(\phi_2)\} = 1 + max\{rank_{R_{\phi_1}}, rank_{R_{\phi_2}}\}$, we have $rank_{R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}} > rank_{R_{\phi_1}}$ and $rank_{R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}} > rank_{R_{\phi_2}}$. According to the definition of stratification, we know that the clause for $\vec{R} \vdash \mathbf{E}[\phi_1 \mathbf{U} \phi_2]$ is also stratified.

Case $\phi = \mathbf{AF}\phi'$: The clause we get consists of two parts, namely the clause generated for $\vec{R} \vdash \phi'$ and $[\forall s : R_{\phi'}(s) \Rightarrow R_{\mathbf{AF}\phi'}(s)] \wedge [\forall s : [\forall s' : \neg T(s,s') \vee R_{\mathbf{AF}\phi'}(s')] \Rightarrow R_{\mathbf{AF}\phi'}(s)]$. According to the induction hypothesis, the clause generated for $\vec{R} \vdash \phi'$ is stratified. Since $rank_{R_{\mathbf{AF}\phi'}} = depth(\mathbf{AF}\phi') = 1 + depth(\phi') = 1 + rank_{R_{\phi'}} > rank_{R_{\phi'}}$ and $rank_T = 0$, therefore according to the definition of stratification, we know that the clause for $\vec{R} \vdash \mathbf{AF}\phi'$ is stratified. $\square$

Theorem 3.2 Given a CTL formula ϕ and an initial interpretation ϱ_0 which defines T and P_p. Assume that ϱ is the least solution to $\vec{R} \vdash \phi \ \wedge \varrho \supseteq \varrho_0$, we have $(M,s) \models \phi$ iff $s \in \varrho(R_\phi)$.

Proof. By Proposition 2.6 we have

$$\varrho = \sqcap\{\varrho | (\varrho, \sigma_0) \models \vec{R} \vdash \phi \wedge \varrho \supseteq \varrho_0\}.$$

We proceed by structural induction on ϕ:

Base cases:

Case $\phi = true$: According to the semantics of CTL, we know that $(M,s) \models true$ for all $s \in S$. From the semantics of ALFP, we know that $\forall s \in S : s \in \varrho(R_{true})$. Therefore, $(M,s) \models true$ iff $s \in \varrho(R_{true})$.

Case $\phi = p$: According to the semantics of CTL, we know that $(M,s) \models p$ iff $p \in L(s)$. From the semantics of ALFP and our assumptions, we know that $s \in \varrho(R_p)$ iff $s \in \varrho(P_p)$ iff $p \in L(s)$. Therefore, $(M,s) \models p$ iff $s \in \varrho(R_p)$.

For boolean connectives:

Case $\phi = \neg\phi'$: Notice that $\varrho(R_{\phi'})$ coincides with $\varrho'(R_{\phi'})$ for the least solution ϱ' to $\vec{R} \vdash \phi' \ \wedge \varrho' \supseteq \varrho_0$. According to the induction hypothesis and our assumptions, we know that $(M,s) \models \phi'$ iff $s \in \varrho(R_{\phi'})$.

From the semantics of ALFP, we know that $s \in \varrho(R_\phi)$ iff $s \notin \varrho(R_{\phi'})$. From the semantics of CTL, we know that $(M, s) \models \phi$ iff $(M, s) \not\models \phi'$. Therefore, $s \in \varrho(R_\phi)$ iff $(M, s) \models \phi$.

Case $\phi = \phi_1 \vee \phi_2$: In this case, it is possible that ϕ_1 and ϕ_2 have a same subformula. We claim that clauses generated for same judgement are the same. Therefore, the same subformula in ϕ_1 and ϕ_2 are dealt with in the same way by flow logic. In the clauses for $\vec{R} \vdash \phi$, we only keep one copy of the clauses for same subformulas in ϕ_1 and ϕ_2. Also notice that $\varrho(R_{\phi_1})$ (resp. $\varrho(R_{\phi_2})$) coincides with $\varrho'(R_{\phi_1})$ (resp. $\varrho'(R_{\phi_2})$) in the least model ϱ' of $\vec{R} \vdash \phi_1 \wedge \varrho' \supseteq \varrho_0$ (resp. $\vec{R} \vdash \phi_2 \wedge \varrho' \supseteq \varrho_0$). According to the induction hypothesis, we know that $s \in \varrho(R_{\phi_1})$ (resp. $s \in \varrho(R_{\phi_2})$) iff $(M, s) \models \phi_1$ (resp. $(M, s) \models \phi_2$).

According the the semantics of CTL, we know that $(M, s) \models \phi_1 \vee \phi_2$ iff $(M, s) \models \phi_1$ or $(M, s) \models \phi_2$. According to the semantics of ALFP, we have $s \in \varrho(R_{\phi_1 \vee \phi_2})$ iff $s \in \varrho(R_{\phi_1})$ or $s \in \varrho(R_{\phi_2})$. Therefore, according to the induction hypothesis and our assumptions, we know that $(M, s) \models \phi_1 \vee \phi_2$ iff $s \in \varrho(R_{\phi_1 \vee \phi_2})$.

For modal operators:

Case $\phi = \mathbf{EX}\phi'$: Notice that $\varrho(R_{\phi'})$ coincides with $\varrho'(R_{\phi'})$ in the least model ϱ' of $\vec{R} \vdash \phi' \wedge \varrho' \supseteq \varrho_0$. According to the induction hypothesis, we know that $(M, s) \models \phi'$ iff $s \in \varrho(R_{\phi'})$.

Since in Kripke structures each state has a successor state, we can extend a finite path fragment $\pi_{fin} = s_0 \rightarrow s_1 \rightarrow ... \rightarrow s_k$ to an infinite path by appending a path starting from s_k to π_{fin}. Similarly, for any path, $s_0 \rightarrow s_1 \rightarrow ...$, its first $k+1$ states forms a finite path fragment of length k. Therefore, according to the semantics of CTL, it is easy to show that $(M, s) \models \mathbf{EX}\phi$ iff there exists a path π from s such that $(M, \pi[1]) \models \phi$ iff $\exists s' : s \rightarrow s' \wedge s' \models \phi'$. From the semantics of ALFP, we have $s \in \varrho(R_{\mathbf{EX}\phi'})$ iff $\exists s' : \varrho(T)(s, s') \wedge s' \in \varrho(R_{\phi'})$. According to the induction hypothesis and our assumptions, we know that $(M, s) \models \mathbf{EX}\phi'$ iff $s \in \varrho(R_{\mathbf{EX}\phi'})$.

Case $\phi = \mathbf{E}[\phi_1 \mathbf{U} \phi_2]$: In this case, it is also possible that ϕ_1 and ϕ_2 have a same subformula. Similarly, we generate same clauses for the same judgement. Therefore, the same subformula in ϕ_1 and ϕ_2 are dealt with in the same way by flow logic. In the clauses for $\mathbf{E}[\phi_1\mathbf{U}\phi_2]$, we only keep one copy of the clauses for same subformulae in ϕ_1 and ϕ_2. Also notice that $\varrho(R_{\phi_1})$ (resp. $\varrho(R_{\phi_2})$) coincides with $\varrho'(R_{\phi_1})$ (resp. $\varrho'(R_{\phi_2})$) in the least model ϱ' of $\vec{R} \vdash \phi_1 \wedge \varrho' \supseteq \varrho_0$ (resp. $\vec{R} \vdash \phi_2 \wedge \varrho' \supseteq \varrho_0$).

From the semantics of ALFP, we know that $\varrho(R_\phi) = \bigcup_K R^K$, where $R^0 = \varrho(R_{\phi_2})$ and $R^K = \{s|\exists s' : \varrho(T)(s,s') \wedge \varrho(R_{\phi_1})(s) \wedge s' \in R^{K-1}\}(K > 0)$. According to our assumptions and the induction hypothesis we get $R^0 = \{s|s \models \phi_2\}$ and $R^K = \{s|\exists s' : s \to s' \wedge s \models \phi_1 \wedge s' \in R^{K-1}\}$.

Since in Kripke structures each state has a successor state, we can extend a finite path fragment $\pi_{fin} = s_0 \to s_1 \to ... \to s_k$ to an infinite path by appending a path starting from s_k to π_{fin}. Similarly, for any path, $s_0 \to s_1 \to ...$, its first $k+1$ states forms a finite path fragment of length k. Therefore, according to the semantics of CTL, it is easy to show that $(M, s) \models \mathbf{E}[\phi_1\mathbf{U}\phi_2]$ iff there exists a path π from s such that $\exists 0 \leq k : (M, \pi[k]) \models \phi_2$ and $\forall\ 0 \leq j < k : (M, \pi[j]) \models \phi_1$ iff there exists a finite path fragment $s_0 \to s_1 \to ... \to s_k$, where $s_0 = s$ and $k \geq 0$ such that $M, s_k \models \phi_2$ and for all $0 \leq i < k, M, s_i \models \phi_1$. Therefore,we can also write $\{s|s \models \phi\} = \bigcup_K S^K$, where $S^K = \{s|$ there exists a finite path fragment $s_0 \to s_1 \to ... \to s_K$, where $s_0 = s$, such that $s_K \models \phi_2$ and $\bigwedge_{0 \leq i < K} s_i \models \phi_1\}$.

We prove $R^K = S^K$ by induction on K.

When $K = 0$, obviously $R^0 = S^0$.

Let's consider $K + 1$. $R^{K+1} = \{s|\exists s' : s \to s' \wedge s \models \phi_1 \wedge s' \in R^K\}$. According to the induction hypothesis, $R^{K+1} = \{s|\exists s' : s \to s' \wedge s \models \phi_1$ and there exists a finite path fragment $s_1 \to s_2 \to ... \to s_{K+1}$, where $s_1 = s'$, such that $s_{K+1} \models \phi_2$ and $\bigwedge_{1 \leq i < K+1} s_i \models \phi_1\}$. Combining $s \to s'$ with the finite path fragment $s_1 \to s_2 \to ... \to s_{K+1}$, we get another finite path fragment $s_0 \to s_1 \to s_2 \to ... \to s_{K+1}$, where $s_0 = s$ and $s_1 = s'$. Therefore, we can also have $R^{K+1} = \{s|$ there exists a finite path fragment $s_0 \to s_1 \to ... \to s_{K+1}$, where $s_0 = s$ and $s_1 = s'$, such that $s_{K+1} \models \phi_2$ and $\bigwedge_{0 \leq i < K+1} s_i \models \phi_1\}$. This is exactly S^{K+1}.

Therefore, we have $s \in \varrho(R_\phi)$ iff $(M, s) \models \phi$.

Case $\phi = \mathbf{AF}\phi'$: Notice that $\varrho(R_\phi)$ coincides with $\varrho'(R_{\phi'})$ in the least model ϱ' of $\vec{R} \vdash \phi' \wedge \varrho' \supseteq \varrho_0$. According to the induction hypothesis, we know that $(M, s) \models \phi'$ iff $s \in \varrho(R_{\phi'})$.

From the semantics of ALFP, we know that $\varrho(R_\phi) = \bigcup_K R^K$, where $R^0 = \varrho(R_{\phi'})$ and $R^{K+1} = \{s|\forall s' : \neg\varrho(T)(s, s') \vee s' \in \bigcup_{k \leq K} R^k\} \cup R^0(K \geq 0)$.

According to our assumptions and the induction hypothesis we get $R^0 = \{s|s \models \phi'\}$. The rest of the proof goes in two steps. We first prove that $\{s|s \models \phi\} = \bigcup_K S^K$, where $S^K = \{s|$ for all finite path fragments $s_0 \to s_1 \to ... \to s_K$, where $s_0 = s$, $\exists k : 0 \leq k \leq K, s_k \models \phi'\}$. Then we will prove by induction on K that $R^K = S^K$.

Now let's proof the first step, that is $\{s|s \models \phi\} = \bigcup_K S^K$. Let's consider the set $T = \{s|s \models \phi\} \setminus \bigcup_K S^K$ and we shall prove that it is empty. We proceed by contradiction. Suppose $T \neq \varnothing$ and choose $s_0 \in T$. It is obvious that $s_0 \models \phi$ but $s_0 \not\models \phi'$ since otherwise $s_0 \in S^0$ (contradicting $s_0 \notin \bigcup_K S^K$). The transition system we consider here is finitely branching, and now we claim that for all successors s_1 of s_0, we have $s_1 \models \phi$. Suppose one successor s_1 of s_0 doesn't satisfy ϕ ($s_1 \not\models \phi$). Then there exists an infinite path starting from s_1 ($s_1 \to s_2 \to ...$) such that for all states along the path, we have $s_i \not\models \phi'(i \geq 1)$. Combining $s_0 \to s_1$ with the infinite path $s_1 \to s_2 \to ...$, we get a new infinite path $s_0 \to s_1 \to s_2 \to ...$ such that for all states along the new path, we have $s_i \not\models \phi'(i \geq 0)$. This means $s_0 \not\models \phi$ and contradicts the fact that $s_0 \in T$. On the other hand, it can't be the case that for all successors s_1 of s_0, we have $s_1 \models \phi'$ since otherwise $s_0 \in S^1$ (contradicting $s_0 \notin \bigcup_K S^K$). We now choose one successor s_1 of s_0 such that $s_1 \models \phi$ but $s_1 \not\models \phi'$. Similarly, we can also show that for all successors s_2 of s_1, we have $s_2 \models \phi$. It can't be the case that for all successors s_2 of s_1, we have $s_2 \models \phi'$ since otherwise $s_0 \in S^2$ (contradicting $s_0 \notin \bigcup_K S^K$). We can choose one successor s_2 of s_1 such that $s_2 \models \phi$ but $s_2 \not\models \phi'$. This process can continue arbitrarily often and produce an infinite path starting from s_0 ($s_0 \to s_1 \to s_2 \to ...$) such that for all the states along the path, we have $s_i \not\models \phi'(i \geq 0)$. This contradicts the assumption $s_0 \models \phi$. Hence $T = \varnothing$.

For the second step, we prove $R^K = S^K$ by induction on K.

When $K = 0$, obviously $R^0 = S^0$.

Let's consider $K+1$. $R^{K+1} = \{s|\forall s' : \neg\varrho(T)(s,s') \vee s' \in \bigcup_{k\leq K} R^k\} \cup R^0$. According to the induction hypothesis, $R^{K+1} = \{s|\forall s' : \neg T(s,s') \vee s' \in \bigcup_{k\leq K} S^k\} \cup S^0$. Clearly, $S^{K+1} = \{s|$ for all finite path fragments $s_0 \rightarrow s_1 \rightarrow ... \rightarrow s_{K+1}$, where $s_0 = s$, $\exists k : 0 \leq k \leq K+1, s_k \models \phi'\} = \{s|$ for all finite path fragments $s_0 \rightarrow s_1 \rightarrow ... \rightarrow s_{K+1}$, where $s_0 = s$, $\exists k : 1 \leq k \leq K+1, s_k \models \phi'\} \cup \{s|s \models \phi'\} = \{s|$ for all s_1 on a finite path fragment $s_0 \rightarrow s_1 \rightarrow ... \rightarrow s_{K+1}$, where $s_0 = s$, we know that for all finite path fragments $s_1 \rightarrow ... \rightarrow s_{K+1}$, $\forall k : 1 \leq k \leq K+1, s_k \models \phi'\} \cup \{s|s \models \phi'\} = \{s|$ for all successor s' of $s, s' \in \bigcup_{k\leq K} S^k\} \cup S^0$.

Therefore, we have $s \in \varrho(R_\phi)$ iff $(M,s) \models \phi$. □

Theorem 3.6 Given a CTL formula ϕ, a fixed CTL fairness assumption `fair`, and an initial interpretation ϱ_0 which defines T and P_p. Assume that ϱ is the least solution to $\vec{R} \vdash_{fair} \phi \ \wedge \varrho \supseteq \varrho_0$ and that $\varrho(Path_{fair,\varphi}) = PATH_{fair,\varrho(R_\varphi)}$ whenever φ is `true` or a subformula of ϕ, we have that $(M,s) \models_{fair} \phi$ iff $s \in \varrho(R_\phi)$.

PROOF. We proceed by structural induction on ϕ.

Case $\phi = true$: According to the semantics of CTL with fairness, we have $\{s|(M,s) \models_{fair} \textbf{true}\} = S$. According to the semantics of ALFP, we know that $\varrho(R_{true}) = S$. Therefore, we know that $(M,s) \models_{fair} \textbf{true}$ iff $s \in \varrho(R_{true})$.

Case $\phi = p$: According to the semantics of CTL with fairness, we have that $\{s|(M,s) \models_{fair} p\} = \{s|p \in L(s)\}$. From the semantics of ALFP and our assumptions, we know that $\varrho(R_p) = \varrho(P_p) = \{s|p \in L(s)\}$. Therefore, we know that $(M,s) \models_{fair} p$ iff $s \in \varrho(R_p)$.

Case $\phi = \neg\phi'$: Notice that $\varrho(R_{\phi'})$ coincides with $\varrho'(R_{\phi'})$ for the least solution ϱ' to $\vec{R} \vdash_{fair} \phi' \ \wedge \varrho' \supseteq \varrho_0$. According to the induction hypothesis and our assumptions, we know that $(M,s) \models_{fair} \phi'$ iff $s \in \varrho(R_{\phi'})$.

From the semantics of ALFP, we know that $s \in \varrho(R_\phi)$ iff $s \notin \varrho(R_{\phi'})$. From the semantics of CTL with fairness, we know that $(M,s) \models_{fair} \phi$ iff $(M,s) \not\models_{fair} \phi'$. Therefore, $s \in \varrho(R_\phi)$ iff $(M,s) \models \phi$.

Case $\phi = \phi_1 \wedge \phi_2$: In this case, it is possible that ϕ_1 and ϕ_2 have a same subformula. We claim that clauses generated for a same judgement are the same. Therefore, the same subformula in ϕ_1 and ϕ_2 are dealt with in the same way by the flow logic. In the clauses for $\vec{R} \vdash_{fair} \phi$, we only keep one copy of the clauses for same subformulae in ϕ_1 and ϕ_2. Also notice that $\varrho(R_{\phi_1})$ (resp. $\varrho(R_{\phi_2})$) coincides with $\varrho'(R_{\phi_1})$ (resp. $\varrho'(R_{\phi_2})$) in the least model ϱ' of $\vec{R} \vdash_{fair} \phi_1 \wedge \varrho' \supseteq \varrho_0$ (resp. $\vec{R} \vdash_{fair} \phi_2 \wedge \varrho' \supseteq \varrho_0$). According to the induction hypothesis, we know that $s \in \varrho(R_{\phi_1})$ (resp. $s \in \varrho(R_{\phi_2})$) iff $(M,s) \models_{fair} \phi_1$ (resp. $(M,s) \models_{fair} \phi_2$).

For the relation $\varrho(R_{\phi_1 \wedge \phi_2})$, it is given by $\varrho(R_{\phi_1 \wedge \phi_2}) = \varrho(R_{\phi_1}) \cap \varrho(R_{\phi_2})$ according to the semantics of ALFP. According to the semantics for CTL with fairness, we know that $\{s|(M,s) \models_{fair} \phi_1\} \cap \{s|(M,s) \models_{fair} \phi_2\} = \{s|(M,s) \models_{fair} \phi_1 \texttt{ and } (M,s) \models_{fair} \phi_2\} = \{s|(M,s) \models_{fair} \phi_1 \wedge \phi_2\}$. Therefore, we have $(M,s) \models_{fair} \phi_1 \wedge \phi_2$ iff $s \in \varrho(R_{\phi_1 \wedge \phi_2})$.

Case $\phi = \mathbf{EX}\phi'$: Notice that $\varrho(R_{\phi'})$ coincides with $\varrho'(R_{\phi'})$ in the least model ϱ' of $\vec{R} \vdash_{fair} \phi' \wedge \varrho' \supseteq \varrho_0$. According to the induction hypothesis, we know that $(M,s) \models_{fair} \phi'$ iff $s \in \varrho(R_{\phi'})$.

For the relation $\varrho(R_{\mathbf{EX}\phi'})$, it is given by $\varrho(R_{\mathbf{EX}\phi'}) = \{s|\exists s' : \varrho(T)(s,s') \wedge \varrho(R_{\phi'})(s') \wedge \varrho(Path_{fair,true})(s')\}$. Notice that, according to our assumptions, $\varrho(Path_{fair,true}) = PATH_{fair,\varrho(R_{true})}$ holds and we have proved above that $\varrho(R_{true}) = \{s|(M,s) \models_{fair} \mathbf{true}\}$. From Lemma 3.4, we have $\varrho(Path_{fair,true}) = \{s|\exists \pi \in Path^{true}_{fair}(s)\}$. According to the assumptions, we know that $\varrho(R_{\mathbf{EX}\phi'}) = \{s|\exists s' : s \to s' \wedge (M,s') \models_{fair} \phi' \wedge \exists \pi \in Path^{true}_{fair}(s')\}$. We now begin to prove that $\varrho(R_{\mathbf{EX}\phi'}) = \{s|\ (M,s) \models_{fair} \mathbf{EX}\phi'\}$.

According to Fact 2.3.2 and the semantics of CTL with fairness, it's obvious that $\{s|\ (M,s) \models_{fair} \mathbf{EX}\phi'\} \subseteq \varrho(R_{\mathbf{EX}\phi'})$. Assume that there exists a transition $s \to s'$ such that $(M,s') \models_{fair} \phi'$ and $\exists \pi' \in Path^{true}_{fair}(s')$. We can extend $s \to s'$ to a path π such that $\pi[0] = s$ and $\pi[i] = \pi'[i-1](i \geq 1)$. According to Fact 2.3.2, π is a fair path. Therefore, $\exists \pi \in Path^{true}_{fair}(s)$ such that $(M,\pi[1]) \models_{fair} \phi'$, which means $(M,s) \models_{fair} \mathbf{EX}\phi'$. This proves the other inclusion $\varrho(R_{\mathbf{EX}\phi'}) \subseteq \{s|\ (M,s) \models_{fair} \mathbf{EX}\phi'\}$. Therefore, we have $(M,s) \models_{fair} \mathbf{EX}\phi'$ iff $s \in \varrho(R_{\mathbf{EX}\phi'})$.

Case $\phi = \mathbf{E}[\phi_1 \mathbf{U} \phi_2]$: In this case, it is also possible that ϕ_1 and ϕ_2 have a same subformula. Similarly, we generate same clauses for the same judgement. Therefore, the same subformula in ϕ_1 and ϕ_2 are dealt with in the same way by flow logic. In the clauses for $\mathbf{E}[\phi_1 \mathbf{U} \phi_2]$, we only keep one copy of the clauses for same subformulae in ϕ_1 and ϕ_2. Also notice that $\varrho(R_{\phi_1})$ (resp. $\varrho(R_{\phi_2})$) coincides with $\varrho'(R_{\phi_1})$ (resp. $\varrho'(R_{\phi_2})$) in the least model ϱ' of $\vec{R} \vdash_{fair} \phi_1 \wedge \varrho' \supseteq \varrho_0$ (resp. $\vec{R} \vdash_{fair} \phi_2 \wedge \varrho' \supseteq \varrho_0$).

For the relation $\varrho(R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]})$, it is given by $\varrho(R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]}) = \bigcup_K R^K$, where $R^0 = \varrho(R_{\phi_2}) \cap \varrho(Path_{fair,true})$ and $R^K = \{s | \exists s' : \varrho(T)(s, s') \wedge \varrho(R_{\phi_1})(s) \wedge s' \in R^{K-1}\} (K > 0)$. Notice that according to our assumptions $\varrho(Path_{fair,true}) = PATH_{fair,\varrho(R_{true})}$ holds and we have proved above that $\varrho(R_{true}) = \{s | (M, s) \models_{fair} \mathbf{true}\}$. From Lemma 3.4, we have $\varrho(Path_{fair,true}) = \{s | \exists \pi \in Path^{true}_{fair}(s)\}$. According to the assumptions and the induction hypothesis, we get $R^0 = \{s | (M, s) \models_{fair} \phi_2 \wedge \exists \pi \in Path^{true}_{fair}(s)\}$ and $R^K = \{s | \exists s' : s \rightarrow s' \wedge (M, s) \models_{fair} \phi_1 \wedge s' \in R^{K-1}\}$. Similarly, we can also write $\{s | (M, s) \models_{fair} \mathbf{E}[\phi_1 \mathbf{U} \phi_2]\} = \bigcup_K S^K$, where $S^K = \{s | \ \exists \pi \in Path^{true}_{fair}(s) : (M, \pi[K]) \models_{fair} \phi_2 \wedge \forall\ 0 \leq j < K : (M, \pi[j]) \models_{fair} \phi_1\}$.

We prove $R^K = S^K$ by induction on K.

The base case is when $K = 0$. It is obvious that $S^0 \subseteq R^0$. Assume that for a state s, we have $(M, s) \models_{fair} \phi_2$ and $\exists \pi \in Path^{true}_{fair}(s)$. It's obvious that $(M, \pi[0]) \models_{fair} \phi_2$. This proves $R^0 \subseteq S^0$.

Let's consider $K + 1$. $R^{K+1} = \{s | \exists s' : s \rightarrow s' \wedge (M, s) \models_{fair} \phi_1 \wedge s' \in R^K\}$. According to the induction hypothesis, we know that $R^{K+1} = \{s | \exists s' : s \rightarrow s' \wedge (M, s) \models_{fair} \phi_1 \wedge s' \in S^K\} = \{s | \exists s' : s \rightarrow s' \wedge (M, s) \models_{fair} \phi_1 \ \wedge \exists \pi \in Path^{true}_{fair}(s') : (M, \pi[K]) \models_{fair} \phi_2 \wedge \forall\ 0 \leq j < K : (M, \pi[j]) \models_{fair} \phi_1\}$. Assume that $s \in R^{K+1}$, we can extend the transition $s \rightarrow s'$ to a path π' by appending the path π to $s \rightarrow s'$ such that $\pi'[0] = s$ and $\pi'[k+1] = \pi[k] (0 \leq k \leq K)$. According to Fact 2.3.2, we know that $\pi' \models fair$. Now we know that $\exists \pi' \in Path^{true}_{fair}(s)$ such that $(M, \pi'[K+1]) \models_{fair} \phi_2 \wedge \forall\ 0 \leq j < K+1 : (M, \pi'[j]) \models_{fair} \phi_1$. Therefore, $s \in S^{K+1}$. This proves $R^{K+1} \subseteq S^{K+1}$.

For the other direction, assume that $s \in S^{K+1}$. Then $\exists \pi \in Path^{true}_{fair}(s)$ such that $(M, \pi[K+1]) \models_{fair} \phi_2 \wedge \forall\ 0 \leq j < K+1 : (M, \pi[j]) \models_{fair} \phi_1$. Consider the suffix π' of the path π such that $\pi'[k] = \pi[k+1] (0 \leq k \leq K)$. According to Fact 2.3.2, π' is a fair path and $(M, \pi'[K]) \models_{fair} \phi_2 \wedge \forall\ 0 \leq j < K : (M, \pi'[j]) \models_{fair} \phi_1$. This means $\pi'[0] \in S^K$ and according to the induction hypothesis we have $\pi'[0] \in R^K$. Therefore, we know that there exists $s'(s' = \pi'[0])$ such that $s \rightarrow s' \wedge (M, s) \models_{fair} \phi_1 \wedge s' \in R^K$. This means $s \in R^{K+1}$. Therefore, we have $S^{K+1} \subseteq R^{K+1}$.

Therefore, we have $(M, s) \models_{fair} \mathbf{E}[\phi_1 \mathbf{U} \phi_2]$ iff $s \in \varrho(R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]})$.

Case $\phi = \mathbf{EG}\phi'$: Notice that $\varrho(R_{\phi'})$ coincides with $\varrho'(R_{\phi'})$ in the least model ϱ' of $\vec{R} \vdash_{fair} \phi' \wedge \varrho' \supseteq \varrho_0$. According to the induction hypothesis, we know that $(M, s) \models_{fair} \phi'$ iff $s \in \varrho(R_{\phi'})$. According to our assumptions, $\varrho(Path_{fair,\phi'}) = PATH_{fair,\varrho(R_{\phi'})}$ holds. From Lemma 3.4, we have $\varrho(Path_{fair,\phi'}) = \{s | \exists \pi \in Path^{\phi'}_{fair}(s)\}$.

The relation $\varrho(R_{\mathbf{EG}\phi'})$ is given by $\varrho(R_{\mathbf{EG}\phi'}) = \varrho(Path_{fair,\phi'})$. It is obvious that $\varrho(Path_{fair,\phi'}) = \{s | \exists \pi : \pi[0] = s \wedge \pi \models fair \wedge \forall 0 \leq i : (M, \pi[i]) \models_{fair} \phi'\} = \{s | (M, s) \models_{fair} \mathbf{EG}\phi'\}$. Therefore, we know that $(M, s) \models_{fair} \mathbf{EG}\phi'$ iff $s \in \varrho(R_{\mathbf{EG}\phi'})$. $\square$

Lemma 3.8 Assume that $ufair = \bigwedge_{1 \leq i \leq k} \mathbf{GF} b_i$. Let C be a nontrivial strongly connected set in M_ϕ such that $C \cap \{s | (M_\phi, s) \models b_i\} \neq \emptyset$ for all $1 \leq i \leq k$. For each state $s \in C$, there exists a path π in M_ϕ from s such that $\pi \models ufair$.

PROOF. For each b_i $(1 \leq i \leq k)$, there exists a state s_i in C such that $(M_\phi, s_i) \models b_i$. Since states $s_1, s_2, ..., s_k$ are mutually reachable from each other, we can construct a cycle in C such that each s_i is visited in this cycle. For example, we can start from s_1 and then visit s_2, s_3 until s_k and finally go back to s_1. Therefore, we can construct an infinite path π from s by first going from s to s_1 and then going around the cycle we have constructed infinitely many times. It is easy to see that $\pi \models ufair$. $\square$

Lemma 3.9 Assume that $ufair = \bigwedge_{1 \leq i \leq k} \mathbf{GF} b_i$. There exists an unconditional fair path in M_ϕ from s if and only if there exists a finite path fragment π_{fin} (in M_ϕ) from s to a state s' in $uFairSCSs_\phi$.

Proof. Assume that we have a finite path fragment π_{fin} from s to a state s' in $uFairSCSs_{\phi}$. From Lemma 3.8, we know that there exists a path π from s' such that $\pi \models ufair$. Therefore, we can construct an unconditional fair path π' by appending π to the end of π_{fin}. From Fact 2.3.2, we know that $\pi' \models ufair$.

Assume that we have an undition fair path π from s and that M_{ϕ} has n states. Since each b_i $(1 \leq i \leq k)$ is satisfied on infinitely many states along π, we can construct a nontrivial strongly connected set C reachable from s such that $C \cap \{s|(M_{\phi}, s) \models b_i\} \neq \emptyset$ for all $1 \leq i \leq k$. We explain it briefly as follows.

The idea is that we want to find a path fragment $\pi_{fragment} = \pi_{fragment}[0], ...,$ $\pi_{fragment}[j]$ $(0 \leq j)$ of π which satisfies the two conditions: 1)$\pi_{fragment}[0] = \pi_{fragment}[j]$ and $(M_{\phi}, \pi_{fragment}[0]) \models b_1$ and 2) for each $2 \leq i \leq k$ there exists a state s'' in $\pi_{fragment}$ such that $(M_{\phi}, s'') \models b_i$. If we can find such a fragment, we can see that $\pi_{fragment}$ forms a cycle and we can construct a nontrivial strongly connected set $C = \{s|s \text{ occurs in } \pi_{fragment}\}$ that is reachable from s.

We will show that it is possible to find such a path fragment. Let's traverse along π from state s. We will first visit a state s_1^1 such that $(M_{\phi}, s_1^1) \models b_1$. Then, we continue along π and will visit a state s_2^1 such that $(M_{\phi}, s_2^1) \models b_2$. We continue this process and will visit a state s_k^1 such that $(M_{\phi}, s_k^1) \models b_k$. Finally, we continue from s_k^1 and will visit a state s_1^2 such that $(M_{\phi}, s_1^2) \models b_1$. The path fragment $\pi_{fragment_1} = s_1^1, ..., s_1^2$ satisfies the following two conditions: 1) $(M_{\phi}, s_1^1) \models b_1$ and $(M_{\phi}, s_1^2) \models b_1$ and 2) for each $2 \leq i \leq k$ there exists a state s'' in $\pi_{fragment_1}$ such that $(M_{\phi}, s'') \models b_i$. If $s_1^2 = s_1^1$, we know that we have found the path fragment we need. Otherwise, we can continue traversing along the path π from s_1^2. We can repeat the above process and find a path fragment $\pi_{fragment_2} = s_1^2, ..., s_1^3$ which satisfies the following two conditions: 1) $(M_{\phi}, s_1^2) \models b_1$ and $(M_{\phi}, s_1^3) \models b_1$ and 2) for each $2 \leq i \leq k$ there exists a state s'' in $\pi_{fragment_2}$ such that $(M_{\phi}, s'') \models b_i$. If $s_1^2 = s_1^3$ or $s_1^1 = s_1^3$, we know that we already find the path fragment we need. Otherwise, we can repeat the above process again.

Assume that there are n states in M_{ϕ} and we have done the above process n times. Then, we have found n path fragments $\pi_{fragment_1}, ..., \pi_{fragment_n}$. Since M_{ϕ} has only n states, at least one of the states $s_1^1, s_1^2, ..., s_1^{n+1}$ has been visited twice in π. Let $s_1^i = s_1^j$ $(1 \leq i, j \leq n+1)$. Then, the path fragment $\pi_{fragment} = s_1^i, ..., s_1^j$ is exactly what we need. $\square$

Lemma 3.10 Let ϱ_0 be an initial interpretation which defines T, P_p and R_ϕ. Assume that $\varrho_0(R_\phi) = \{s|(M,s) \models_{ufair} \phi\}$. For the least solution ϱ to $\vec{R} \Vdash Path_{ufair,\phi} \wedge \varrho \supseteq \varrho_0$, we have the following:

- $\varrho(T_\phi)$ equals the transition relation in M_ϕ,
- $(s,s') \in \varrho(T_\phi^+)$ iff there exists a finite path fragment $\pi_{fin} = s_0, s_1...s_n$ in M_ϕ where $s_0 = s$ and $s_n = s'$,
- $(s,s') \in \varrho(SC_\phi)$ iff s and s' belong to a nontrivial strongly connected set in M_ϕ,
- $\varrho(SC_{ufair,\phi}) = uFairSCSs_\phi$, and
- $\varrho(Path_{ufair,\phi}) = \{s|\exists\pi : \pi \in Path^\phi_{ufair}(s)\}$.

PROOF. First, we want to prove that $\varrho(T_\phi)$ equals the transition relation in M_ϕ. From the semantics of ALFP, we know that $(s,s') \in \varrho(T_\phi)$ if and only if $(s,s') \in \varrho(T)$ and $s, s' \in \varrho(R_\phi)$. Since ϱ_0 defines T and $\varrho_0(R_\phi) = \{s|(M,s) \models_{ufair} \phi\}$, we know that $(s,s') \in \varrho(T_\phi)$ if and only if $(s,s') \in T$ and $(M,s) \models_{ufair} \phi$ and $(M,s') \models_{ufair} \phi$. Therefore, according to the definition of M_ϕ, we know that $\varrho(T_\phi)$ equals the transition relation in M_ϕ.

The statement for the relation $\varrho(T_\phi^+)$ is obvious since T_ϕ^+ is actually the transitive closure of T_ϕ.

We now prove that $(s,s') \in \varrho(SC_\phi)$ if and only if s and s' belong to a nontrivial strongly connected set in M_ϕ. The relation $\varrho(SC_\phi)$ is given by $\varrho(SC_\phi) = \{(s,s')|\varrho(T_\phi^+)(s,s') \wedge \varrho(T_\phi^+)(s',s)\}$. From above, we know that $\varrho(SC_\phi) = \{(s,s')|$ there is a finite path π_{fin} from s to s' and a finite path π'_{fin} from s' to s in $M_\phi\}$.

Assume that s and s' belong to a nontrivial strongly connected set in M_ϕ, then we know that there is a finite path π_{fin} from s to s' and a finite path π'_{fin} from s' to s in M_ϕ. Then we have $(s,s') \in \varrho(SC_\phi)$. This proves one direction. Assume that $(s,s') \in \varrho(SC_\phi)$, then there is a finite path π_{fin} from s to s' and a finite path π'_{fin} from s' to s in M_ϕ. Then, the set $C' = \{s|\ s$ is a state on π_{fin} or $\pi'_{fin}\}$ is a nontrivial strongly connected set in M_ϕ. Since $s, s' \in C'$, we know that the other direction holds.

Let us prove that $\varrho(SC_{ufair,\phi}) = uFairSCSs_\phi$. The relation $\varrho(SC_{ufair,\phi})$ is given by $\varrho(SC_{ufair,\phi}) = \{s|\forall 1 \leq i \leq k : \exists s_i : \varrho(SC_\phi)(s, s_i) \wedge \varrho(P_{b_i})(s_i)\}$. Since $\varrho(P_{b_i}) = \{s|(M, s) \models b_i\}$, we know that $uFairSCSs_\phi$ is actually the set union of all nontrivial strongly connected sets C, in M_ϕ, satisfying $C \cap \varrho(P_{b_i}) \neq \emptyset$ for all $1 \leq i \leq k$.

Assume that $s \in uFairSCSs_\phi$, then s belongs to a nontrivial strongly connected set C satisfying $C \cap \varrho(P_{b_i}) \neq \emptyset$ for all $1 \leq i \leq k$. Therefore, for each $1 \leq i \leq k$, there exists a state $s_i \in C$ such that $s_i \in \varrho(P_{b_i})$. Since s and s_i are in the same nontrivial strongly connected set C, we have $(s, s_i) \in \varrho(SC_\phi)$. Therefore, we know that $\forall 1 \leq i \leq k : \exists s_i : \varrho(SC_\phi)(s, s_i) \wedge \varrho(P_{b_i})(s_i)$. This means $s \in \varrho(SC_{ufair,\phi})$. This proves one direction.

Assume that $s \in \varrho(SC_{ufair,\phi})$, then $\forall 1 \leq i \leq k : \exists s_i : \varrho(SC_\phi)(s, s_i) \wedge \varrho(P_{b_i})(s_i)$. Since, for each $1 \leq i \leq k$, $(s, s_i) \in \varrho(SC_\phi)$ holds, there exists a nontrivial strongly connected set C such that, for all $1 \leq i \leq k$, s and s_i belong to C. Since $s_i \in \varrho(P_{b_i})$, we know that $C \cap \varrho(P_{b_i}) \neq \emptyset$. Therefore, $s \in uFairSCSs_\phi$. This proves the other direction.

We then prove that $\varrho(Path_{ufair,\phi}) = \{s|\exists\pi : \pi \in Path^{\phi}_{ufair}(s)\}$. We want to show that $s \in \varrho(Path_{ufair,\phi})$ iff there exists an unconditional fair path π in M_ϕ from s. According to ALFP semantics, the relation $\varrho(Path_{ufair,\phi})$ is given by $\varrho(Path_{ufair,\phi}) = \{s|\exists s' : \varrho(T^+_\phi)(s, s') \wedge \varrho(SC_{ufair,\phi})(s')\}$. From above, we know that $\varrho(Path_{ufair,\phi}) = \{s|$ there exists a finite fragment π_{fin} in M_ϕ from s to a state s' in $uFairSCSs_\phi\}$. According to Lemma 3.9, we know that $s \in \varrho(Path_{ufair,\phi})$ iff there exists an unconditional fair path π in M_ϕ from s. Therefore, we have $\varrho(Path_{ufair,\phi}) = \{s|\exists\pi : \pi \in Path^{\phi}_{ufair}(s)\}$. $\square$

Lemma 3.12 Assume that $sfair = \mathbf{GF}a \Rightarrow \mathbf{GF}b$. Let C be a nontrivial strongly connected set in M_ϕ such that either $C \cap \{s|(M_\phi, s) \models b\} \neq \emptyset$ or $\forall s \in C : \{s|(M_\phi, s) \not\models a\}$. For each state $s \in C$, there exists a path π in M_ϕ from s such that $\pi \models sfair$.

PROOF. Assume that $C \cap \{s|(M_\phi, s) \models b\} \neq \emptyset$ holds. Let s' be a state in C such that $(M_\phi, s') \models b$. We can find a finite path fragment π_{fin} from s to s' and another path fragment π'_{fin} from s' to s. The two path fragments form a cycle. Therefore, starting from s we could go around the cycle we have constructed infinitely many times. This forms an infinite path π from s such that b is satisfied on infinitely many states in π. Therefore, $\pi \models sfair$.

Assume that $\forall s \in C : \{s | (M_\phi, s) \nvDash a\}$. Let s' be a state in C. We can find a finite path fragment π_{fin} from s to s' and another path fragment π'_{fin} from s' to s. The two path fragments form a cycle. Therefore, starting from s we could go around the cycle we have constructed infinitely many times. This forms an infinite path π from s such that a is not satisfied on any of the states in π. Therefore, $\pi \models sfair$. □

Lemma 3.13 Assume that $sfair = \mathbf{GF}a \Rightarrow \mathbf{GF}b$. There exists a strong fair path in M_ϕ from s if and only if there exists a finite path fragment π_{fin}, in M_ϕ, from s to a state s' in $sFairSCSs_\phi$.

Proof. Assume that we have a finite path fragment π_{fin} from s to a state s' in $sFairSCSs_\phi$. From Lemma 3.12, we know that there exists a path π from s' such that $\pi \models sfair$. Therefore, we can construct a strong fair path π' by appending π to the end of π_{fin}. From Fact 2.3.2, we know that $\pi' \models sfair$.

Assume that π is a strong fair path from s and that M_ϕ has n states. There are two cases such that $sfair$ is satisfied on π.

The first case is that a is satisfied only on finitely many states in π. In this case, there exists a suffix π' of π such that a is not satisfied on any of the states in π'. In the prefix $\pi'_{fin} = s_0, ..., s_n$ of π', we know that at least one state has been visited twice since M_ϕ has only n states. Assume that s' has been visited twice in π'_{fin}. Then, the finite path fragment $\pi''_{fin} = s_i, ..., s_j$ in π'_{fin} such that $s_i = s_j = s'$ $(0 \leq i, j \leq n)$ forms a cycle. We can thus construct a nontrivial strongly connected set $C = \{s | s \text{ occurs in } \pi''_{fin}\}$ that is reachable from s.

The second case is that b is satisfied on infinitely many states in π. Let $\pi_{fragment}$ be a path fragment in π such that b is satisfied on $n+1$ states in $\pi_{fragment}$. We know that at least one of these $n+1$ states has been visited twice (in $\pi_{fragment}$) since M_ϕ has only n states. Assume that s' is one of these $n+1$ states that has been visited twice in $\pi_{fragment}$. Then, the finite path fragment $\pi'_{fin} = s_i, ..., s_j$ in π'_{fin} such that $s_i = s_j = s'$ $(0 \leq i, j)$ forms a cycle. We can thus construct a nontrivial strongly connected set $C = \{s | s \text{ occurs in } \pi'_{fin}\}$ that is reachable from s. □

Lemma 3.14 Let ϱ_0 be an initial interpretation which defines T, P_p, $R_{\neg a}$ and R_ϕ. Assume that $\varrho_0(R_\phi) = \{s|(M,s) \models_{sfair} \phi\}$ and $\varrho_0(R_{\neg a}) = \{s|(M,s) \not\models a\}$. For the least solution ϱ to $\vec{R} \Vdash Path_{sfair,\phi} \wedge \varrho \supseteq \varrho_0$, we have the following:

- $\varrho(T_\phi)$ (resp. $\varrho(T_{\phi\wedge\neg a})$) equals the transition relation in M_ϕ (resp. $M_{\phi\wedge\neg a}$),
- $(s,s') \in \varrho(T^+_\phi)$ (resp. $(s,s') \in \varrho(T^+_{\phi\wedge\neg a})$) iff there exists a finite path fragment $\pi_{fin} = s_0, s_1 ... s_n$ in M_ϕ (resp. $M_{\phi\wedge\neg a}$) where $s_0 = s$ and $s_n = s'$,
- $(s,s') \in \varrho(SC_\phi)$ (resp. $(s,s') \in \varrho(SC_{\phi\wedge\neg a})$) iff s and s' belong to a nontrivial strongly connected set in M_ϕ (resp. $M_{\phi\wedge\neg a}$),
- $\varrho(SC_{sfair,\phi}) = sFairSCSs_\phi$, and
- $\varrho(Path_{sfair,\phi}) = \{s|\exists\pi : \pi \in Path^{\phi}_{sfair}(s)\}$.

Proof. Proofs for the cases of $\varrho(T_\phi)$, $\varrho(T^+_\phi)$ and $\varrho(SC_\phi)$ are the same as those given in the proof of Lemma 3.10. We can proof the cases of $\varrho(T_{\phi\wedge\neg a})$, $\varrho(T^+_{\phi\wedge\neg a})$ and $\varrho(SC_{\phi\wedge\neg a})$ similarly.

Now we prove that $\varrho(SC_{sfair,\phi}) = sFairSCSs_\phi$. Since $\varrho(P_a) = \{s|(M,s) \models a\}$ and $\varrho(P_b) = \{s|(M,s) \models b\}$, $sFairSCSs_\phi$ is actually the set union of all nontrivial strongly connected sets C, in M_ϕ, satisfying either $C \cap \varrho(P_b) \neq \emptyset$ or $\forall s \in C : s \notin \varrho(P_a)$.

According to the semantics of ALFP, the relation $\varrho(SC_{sfair,\phi})$ is given by $\varrho(SC_{sfair,\phi}) = \{s|\exists s' : \varrho(SC_\phi)(s,s') \wedge \varrho(P_b)(s')\} \cup \{s|\exists s' : \varrho(SC_{\phi\wedge\neg a})(s,s')\}$.

Similar with the proof for Lemma 3.10, we know that the set $\{s|\exists s' : \varrho(SC_\phi)(s,s') \wedge \varrho(P_b)(s')\}$ equals the set union of all nontrivial strongly connected sets C, in M_ϕ, satisfying $C \cap \varrho(P_b) \neq \emptyset$, and the set $\{s|\exists s' : \varrho(SC_{\phi\wedge\neg a})(s,s')\}$ equals the set union of all nontrivial strongly connected sets C, in $M_{\phi\wedge\neg a}$. In the following, we want to prove that the set $\{s|\exists s' : \varrho(SC_{\phi\wedge\neg a})(s,s')\}$ equals the set union of all nontrivial strongly connected sets C, in M_ϕ, satisfying $\forall s \in C : \{s|(M_\phi,s) \not\models a\}$.

Notice that the transition graph in $M_{\phi\wedge\neg a}$ is a subgraph of that in M_ϕ. Therefore, a nontrivial strongly connected set C in $M_{\phi\wedge\neg a}$ is also a nontrivial strongly connected set in M_ϕ satisfying $\forall s \in C : \{s|(M_\phi,s) \not\models a\}$. This proves one direction.

Assume C is a nontrivial strongly connected set in M_ϕ satisfying $\forall s \in C : \{s|(M_\phi, s) \nvDash a\}$. Notice that $(M_\phi, s) \nvDash a$ iff $(M_\phi, s) \nvDash_{sfair} a$ since a is a proposition here. For any two states s and s' in C, we know that there exists a finite path π_{fin} in M_ϕ from s to s' such that $\forall 0 \leq i < |\pi_{fin}| : (M_\phi, \pi_{fin}[i]) \nvDash_{sfair} a$ and that there exists a finite path π'_{fin} in M_ϕ from s' to s such that $\forall 0 \leq i < |\pi'_{fin}| : (M_\phi, \pi'_{fin}[i]) \nvDash_{sfair} a$. It's obvious that both π_{fin} and π'_{fin} are valid paths in $M_{\phi\wedge\neg a}$. Therefore, s and s' belong to a nontrivial strongly connected set in $M_{\phi\wedge\neg a}$. Therefore, C is a nontrivial strongly connected set in $M_{\phi\wedge\neg a}$. This proves the other direction.

From above, we know that $\varrho(SC_{sfair,\phi})$ equals the set union of all nontrivial strongly connected sets C, in M_ϕ, satisfying either $C \cap \{s|(M, s) \models b\} \neq \emptyset$ or $\forall s \in C : \{s|(M_\phi, s) \nvDash a\}$. From the definition of $sFairSCSs_\phi$, we have $\varrho(SC_{sfair,\phi}) = sFairSCSs_\phi$.

We now prove $\varrho(Path_{sfair,\phi}) = \{s|\exists\pi : \pi \in Path^\phi_{sfair}(s)\}$. We prove this by showing that $s \in \varrho(Path_{sfair,\phi})$ iff there exists a strong fair path π in M_ϕ from s. The relation $\varrho(Path_{sfair,\phi})$ is given by $\varrho(Path_{sfair,\phi}) = \{s|\exists s' : \varrho(T^+_\phi)(s, s') \wedge \varrho(SC_{sfair,\phi})(s')\}$. From above, we know that $\varrho(Path_{sfair,\phi}) = \{s|$ there exists a finite fragment π_{fin} in M_ϕ from s to a state s' in $sFairSCSs_\phi\}$. According to Lemma 3.13, we know that $s \in \varrho(Path_{sfair,\phi})$ iff there exists a strong fair path π in M_ϕ from s. Therefore, we have $\varrho(Path_{sfair,\phi}) = \{s|\exists\pi : \pi \in Path^\phi_{sfair}(s)\}$.
$\square$

Appendix B

Appendix for Chapter 4

Theorem 4.2 $\{\varrho | [(\varrho, \sigma_0) \ \underline{\texttt{sat}} \ cl] = true\}$ is a Moore Family with respect to $\sqsubseteq_\sharp$, i.e. is closed under greatest lower bounds, whenever cl is closed and stratified; the greatest lower bound $\sqcap_\sharp \ \{\varrho | [(\varrho, \sigma_0) \ \underline{\texttt{sat}} \ cl] = true\}$ is the *least* model of cl.

More generally, given ϱ_0 the set $\{\varrho | [(\varrho, \sigma_0) \ \underline{\texttt{sat}} \ cl] = true \wedge \varrho_0 \sqsubseteq \varrho\}$ is a Moore Family with respect to $\sqsubseteq_\sharp$ and $\sqcap_\sharp \ \{\varrho | [(\varrho, \sigma_0) \ \underline{\texttt{sat}} \ cl] = true \wedge \varrho_0 \sqsubseteq \varrho\}$ is the least model.

Proof. Assume cl has the form $cl_1 \wedge ... \wedge cl_s$ where cl_j is the clause corresponding to stratum j, and let $\mathcal{R}_j$ denote the set of all relation symbols R defined in $cl_1 \wedge ... \wedge cl_j$ and ϱ_0. Let $\mathcal{R}_0$ denote the set of all relation symbols defined in ϱ_0. Let M denote a set of assignments which maps relation symbols to a multi-valued function. Then $\varrho = \sqcap_\sharp M$ is defined by the formula

$$\varrho(R) = \bigsqcap \{\varrho'(R) | \varrho' \in M \wedge \forall R' \in \mathcal{R}_{rank(R)-1} : \varrho(R') = \varrho'(R')\},$$

which is well-defined by induction on the value of $rank(R)$.

The theorem holds from the fact that for all j, all M and all variable environment σ Lemma B.1 and Lemma B.2 hold. $\square$

Lemma B.1 *Assume that* $\varrho = \sqcap_\sharp M$ *and* pre *occurs in* cl_j. *Let* $M_j = \{\varrho' \in M | \forall R' \in \mathcal{R}_{j-1} : \varrho(R') = \varrho'(R')\}$. *We know that* $[(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre] \sqsubseteq [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre]$ *for all* $\varrho' \in M_j$.

Proof. We proceed by induction on j and in each case perform a structural induction on pre occurring in cl_j.

Case $pre = R(v_1, ..., v_n)$: Let $\varrho' \in M_j$. According to the semantics of ALFP, we have $[(\varrho, \sigma)\ \underline{\mathtt{sat}}\ R(v_1, ..., v_n)] = \varrho(R)(\sigma(v_1), ..., \sigma(v_n))$ and $[(\varrho', \sigma)\ \underline{\mathtt{sat}}\ R(v_1, ..., v_n)] = \varrho'(R)(\sigma(v_1), ..., \sigma(v_n))$. According to the definition of ϱ and ϱ', we know that $\varrho(R) \sqsubseteq \varrho'(R)$. Hence, we know that $[(\varrho, \sigma)\ \underline{\mathtt{sat}}\ R(v_1, ..., v_n)] \sqsubseteq [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ R(v_1, ..., v_n)]$. Therefore, we know that $[(\varrho, \sigma)\ \underline{\mathtt{sat}}\ R(v_1, ..., v_n)] \sqsubseteq [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ R(v_1, ..., v_n)]$ for all $\varrho' \in M_j$.

Case $pre = \neg R(v_1, ..., v_n)$: According to stratification, we know that $rank(R) < j$ and hence $\varrho'(R) = \varrho(R)$ for all $\varrho' \in M_j$. Therefore, the result holds.

Case $pre = pre_1 \vee pre_2$: Let $\varrho' \in M_j$. According to the semantics of ALFP, we have $[(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1 \vee pre_2] = [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1] \sqcup [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_2]$ and $[(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre_1 \vee pre_2] = [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre_1] \sqcup [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre_2]$. According to the induction hypothesis, we know that $[(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1] \sqsubseteq [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre_1]$ and $[(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_2] \sqsubseteq [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre_2]$. Therefore, we have the following: $[(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1 \vee pre_2] = [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1] \sqcup [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_2] \sqsubseteq [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre_1] \sqcup [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre_2] = [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre_1 \vee pre_2]$. Therefore, we know that $[(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1 \vee pre_2] \sqsubseteq [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre_1 \vee pre_2]$ for all $\varrho' \in M_j$.

Case $pre = pre_1 \wedge pre_2$: Let $\varrho' \in M_j$. According to the semantics of ALFP, we have $[(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1 \wedge pre_2] = [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1] \sqcap [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_2]$ and $[(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre_1 \wedge pre_2] = [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre_1] \sqcap [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre_2]$. According to the induction hypothesis, we know that $[(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1] \sqsubseteq [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre_1]$ and $[(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_2] \sqsubseteq [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre_2]$. Therefore, we have the following: $[(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1 \wedge pre_2] = [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1] \sqcap [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_2] \sqsubseteq [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre_1] \sqcap [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre_2] = [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre_1 \wedge pre_2]$. Therefore, we know that $[(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1 \wedge pre_2] \sqsubseteq [(\varrho', \sigma)\ \underline{\mathtt{sat}}\ pre_1 \wedge pre_2]$ for all $\varrho' \in M_j$.

Case $pre = \exists x :\ pre'$: Let $\varrho' \in M_j$. According to the semantics of ALFP, we have $[(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \exists x :\ pre'] = \bigsqcup_{a \in \mathcal{U}}\{[(\varrho, \sigma[x \mapsto a])\ \underline{\mathtt{sat}}\ pre']\}$ and $[(\varrho', \sigma)\ \underline{\mathtt{sat}}\ \exists x :\ pre'] = \bigsqcup_{a \in \mathcal{U}}\{[(\varrho', \sigma[x \mapsto a])\ \underline{\mathtt{sat}}\ pre']\}$. According to the induction hypothesis, we know that $\forall a \in \mathcal{U} : [(\varrho, \sigma[x \mapsto a]) \;\underline{\texttt{sat}}\; pre'] \sqsubseteq [(\varrho', \sigma[x \mapsto a]) \;\underline{\texttt{sat}}\; pre']$. Therefore, we have the following: $[(\varrho, \sigma) \;\underline{\texttt{sat}}\; \exists x : pre'] = \bigsqcup_{a \in \mathcal{U}} \{[(\varrho, \sigma[x \mapsto a]) \;\underline{\texttt{sat}}\; pre']\} \sqsubseteq \bigsqcup_{a \in \mathcal{U}} \{[(\varrho', \sigma[x \mapsto a]) \;\underline{\texttt{sat}}\; pre']\} = [(\varrho', \sigma) \;\underline{\texttt{sat}}\; \exists x : pre']$. Therefore, we know that $[(\varrho, \sigma) \;\underline{\texttt{sat}}\; \exists x : pre'] \sqsubseteq [(\varrho', \sigma) \;\underline{\texttt{sat}}\; \exists x : pre']$ for all $\varrho' \in M_j$.

Case $pre = \forall x : pre'$: Let $\varrho' \in M_j$. According to the semantics of ALFP, we have $[(\varrho, \sigma) \;\underline{\texttt{sat}}\; \forall x : pre'] = \bigsqcap_{a \in \mathcal{U}} \{[(\varrho, \sigma[x \mapsto a]) \;\underline{\texttt{sat}}\; pre']\}$ and $[(\varrho', \sigma) \;\underline{\texttt{sat}}\; \forall x : pre'] = \bigsqcap_{a \in \mathcal{U}} \{[(\varrho', \sigma[x \mapsto a]) \;\underline{\texttt{sat}}\; pre']\}$. According to the induction hypothesis, we know that $\forall a \in \mathcal{U} : [(\varrho, \sigma[x \mapsto a]) \;\underline{\texttt{sat}}\; pre'] \sqsubseteq [(\varrho', \sigma[x \mapsto a]) \;\underline{\texttt{sat}}\; pre']$. Therefore, we have the following: $[(\varrho, \sigma) \;\underline{\texttt{sat}}\; \forall x : pre'] = \bigsqcap_{a \in \mathcal{U}} \{[(\varrho, \sigma[x \mapsto a]) \;\underline{\texttt{sat}}\; pre']\} \sqsubseteq \bigsqcap_{a \in \mathcal{U}} \{[(\varrho', \sigma[x \mapsto a]) \;\underline{\texttt{sat}}\; pre']\} = [(\varrho', \sigma) \;\underline{\texttt{sat}}\; \forall x : pre']$. Therefore, we know that $[(\varrho, \sigma) \;\underline{\texttt{sat}}\; \forall x : pre'] \sqsubseteq [(\varrho', \sigma) \;\underline{\texttt{sat}}\; \forall x : pre']$ for all $\varrho' \in M_j$. □

Lemma B.2 *Assume that $\varrho = \sqcap_\sharp M$ and cl occurs in cl_j. If $[(\varrho', \sigma) \;\underline{\texttt{sat}}\; cl] = true$ for all $\varrho' \in M$, then $[(\varrho, \sigma) \;\underline{\texttt{sat}}\; cl] = true$.*

Proof. We proof by induction on j and in each case distinguish between two cases. The first case is when $\forall x \in \mathcal{U}^k : \varrho(R)(x) = \top$ for all relations R of rank j and arity k. It is straightforward by induction on cl to show that our lemma holds. The second case is when $\exists x \in \mathcal{U}^k : \varrho(R)(x) \neq \top$ for some relation R of rank j and arity k. Then the set

$$M_j = \{\varrho' \in M | \forall R' \in \mathcal{R}_{j-1} : \varrho(R') = \varrho'(R')\}$$

is not empty. Therefore, we have that $\varrho(R) = \sqcap\{\varrho'(R) | \varrho' \in M_j\}$ if $rank_R = j$ and that $\varrho(R) = \varrho'(R)$ if $rank_R < j$ and $\varrho' \in M_j$. We proceed by structural induction on cl occurring cl_j.

Case $cl = \mathbf{true}$: It's obvious that $[(\varrho, \sigma) \;\underline{\texttt{sat}}\; \mathbf{true}] = true$.

Case $cl = cl_1 \wedge cl_2$: According to the semantics of ALFP, we have $[(\varrho, \sigma) \;\underline{\texttt{sat}}\; cl_1 \wedge cl_2] = [(\varrho, \sigma) \;\underline{\texttt{sat}}\; cl_1] \wedge [(\varrho, \sigma) \;\underline{\texttt{sat}}\; cl_2]$. If $[(\varrho', \sigma) \;\underline{\texttt{sat}}\; cl_1 \wedge cl_2] = true$ for all $\varrho' \in M_j$, we know that $[(\varrho', \sigma) \;\underline{\texttt{sat}}\; cl_1] = true$ and $[(\varrho', \sigma) \;\underline{\texttt{sat}}\; cl_2] = true$ for all $\varrho' \in M_j$. According to the induction hypothesis, we have $[(\varrho, \sigma) \;\underline{\texttt{sat}}\; cl_1] = true$ and $[(\varrho, \sigma) \;\underline{\texttt{sat}}\; cl_2] = true$. Therefore, $[(\varrho, \sigma) \;\underline{\texttt{sat}}\; cl_1 \wedge cl_2] = true$.

Case $cl = \forall x : cl'$: According to the semantics of ALFP, we know that $[(\varrho,\sigma)\ \underline{\mathtt{sat}}\ \forall x :\ cl'] = true$ iff $\forall a \in \mathcal{U} : [(\varrho,\sigma[x \mapsto a])\ \underline{\mathtt{sat}}\ cl'] = true$. If $[(\varrho',\sigma)\ \underline{\mathtt{sat}}\ \forall x :\ cl'] = true$ for all $\varrho' \in M_j$, we must have $\forall a \in \mathcal{U} : [(\varrho',\sigma[x \mapsto a])\ \underline{\mathtt{sat}}\ cl'] = true$ for all $\varrho' \in M_j$. According to the induction hypothesis, we have $\forall a \in \mathcal{U} : [(\varrho,\sigma[x \mapsto a])\ \underline{\mathtt{sat}}\ cl'] = true$. Therefore, $[(\varrho,\sigma)\ \underline{\mathtt{sat}}\ \forall x :\ cl'] = true$.

Case $cl = pre \Rightarrow R(v_1,...,v_n)$: According to the semantics of ALFP, we know that $[(\varrho,\sigma)\ \underline{\mathtt{sat}}\ pre \Rightarrow R(v_1,...,v_n)] = true$ iff $[(\varrho,\sigma)\ \underline{\mathtt{sat}}\ pre] \sqsubseteq [(\varrho,\sigma)\ \underline{\mathtt{sat}}\ R(v_1,...,v_n)]$. If $[(\varrho',\sigma)\ \underline{\mathtt{sat}}\ pre \Rightarrow R(v_1,...,v_n)] = true$ for all $\varrho' \in M_j$, we know that $[(\varrho',\sigma)\ \underline{\mathtt{sat}}\ pre] \sqsubseteq [(\varrho',\sigma)\ \underline{\mathtt{sat}}\ R(v_1,...,v_n)]$ for all $\varrho' \in M_j$. Therefore, $\sqcap\{[(\varrho',\sigma)\ \underline{\mathtt{sat}}\ pre] | \varrho' \in M_j\} \sqsubseteq \sqcap\{[(\varrho',\sigma)\ \underline{\mathtt{sat}}\ R(v_1,...,v_n)] | \varrho' \in M_j\}$.

From Lemma B.1, we know that $[(\varrho,\sigma)\ \underline{\mathtt{sat}}\ pre] \sqsubseteq [(\varrho',\sigma)\ \underline{\mathtt{sat}}\ pre]$. Therefore, $[(\varrho,\sigma)\ \underline{\mathtt{sat}}\ pre] \sqsubseteq \sqcap\{[(\varrho',\sigma)\ \underline{\mathtt{sat}}\ pre] | \varrho' \in M_j\}$. According to the definition of ϱ and ϱ', we know that $[(\varrho,\sigma)\ \underline{\mathtt{sat}}\ R(v_1,...,v_n)] = \sqcap\{[(\varrho',\sigma)\ \underline{\mathtt{sat}}\ R(v_1,...,v_n)] | \varrho' \in M_j\}$. Therefore, $[(\varrho,\sigma)\ \underline{\mathtt{sat}}\ pre] \sqsubseteq \sqcap\{[(\varrho',\sigma)\ \underline{\mathtt{sat}}\ pre] | \varrho' \in M_j\} \sqsubseteq \sqcap\{[(\varrho',\sigma)\ \underline{\mathtt{sat}}\ R(v_1,...,v_n)] | \varrho' \in M_j\} = [(\varrho,\sigma)\ \underline{\mathtt{sat}}\ R(v_1,...,v_n)]$. This means $[(\varrho,\sigma)\ \underline{\mathtt{sat}}\ pre \Rightarrow R(v_1,...,v_n)] = true$. □

From [59] we have Proposition B.3 and Lemma B.4.

Proposition B.3 *Let $\mathcal{L} = (L, \sqsubseteq)$ be a finite lattice. For all $x \in L$, we know that $x = \bigsqcup\{y | y \in \mathcal{J}(\mathcal{L}), y \sqsubseteq x\}$.*

Lemma B.4 *Let $\mathcal{L} = (L, \sqsubseteq)$ be a distributive lattice and $x \in \mathcal{J}(\mathcal{L})$. We know that for any $1 \leq k$, if $y_1,...,y_k \in L$ and $x \sqsubseteq y_1 \sqcup ... \sqcup y_k$, then $x \sqsubseteq y_i$ for some $1 \leq i \leq k$.*

Lemma B.5 *Let $\mathcal{M} = (\mathcal{L}, \sim)$ be a finite distributive multi-valued structure and $\mathcal{J}(\mathcal{L}) = \{x_1,...,x_n\}$. Then we have: $\mathcal{C}((\varrho^{x_1},...,\varrho^{x_n}))$ iff $x_j \in \mathcal{J}(\mathcal{L})$ and $x_j \sqsubseteq \bigsqcup\{x_i | \varrho^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\}$ implies $\varrho^{x_j}(R)(s) = true$ where $s \in \mathcal{U}^k, R \in \mathcal{R}$ and $1 \leq j \leq n$.*

Proof. Assume that $x_j \in \mathcal{J}(\mathcal{L})$ and $x_j \sqsubseteq \bigsqcup\{x_i | \varrho^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\}$ implies $\varrho^{x_j}(R)(s) = true$ where $s \in \mathcal{U}^k, R \in \mathcal{R}$ and $1 \leq j \leq n$. Assume that $x_i \sqsupseteq x_j$ and that $\varrho^{x_i}(R)(s) = true$. Therefore, we have $x_j \sqsubseteq x_i \sqsubseteq \bigsqcup\{x_i | \varrho^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\}$. According to our assumption, we know that $\varrho^{x_j}(R)(s) = true$. Since s and R are arbitrarily chosen, we know that $\varrho^{x_i} \sqsubseteq \varrho^{x_j}$.

Assume that $\mathcal{C}((\varrho^{x_1},...,\varrho^{x_n}))$ holds, which means $x_i \sqsupseteq x_j$ implies $\varrho^{x_i} \sqsubseteq \varrho^{x_j}$. Let $s \in \mathcal{U}^k$ and $R \in \mathcal{R}$. Assume that $x_j \sqsubseteq \bigsqcup\{x_i|\varrho^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\}$. Since $x_j \sqsubseteq \bigsqcup\{x_i|\varrho^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\}$ iff $x_j = x_j \sqcap \bigsqcup\{x_i|\varrho^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\}$, we know that $x_j = x_j \sqcap \bigsqcup\{x_i|\varrho^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\}$ holds. Since $\mathcal{L}$ is distributive, we know that $x_j = x_j \sqcap \bigsqcup\{x_i|\varrho^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\} = \bigsqcup\{x_j \sqcap x_i|\varrho^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\}$. Therefore, $x_j \sqsubseteq \bigsqcup\{x_j \sqcap x_i|\varrho^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\}$. From Lemma B.4, we know that $x_j \sqsubseteq x_j \sqcap x_i$ for some x_i such that $\varrho^{x_i}(R)(s) = true$. This means $x_j \sqsubseteq x_i$. From our assumption, we know that $\varrho^{x_i} \sqsubseteq \varrho^{x_j}$. Therefore, $\varrho^{x_j}(R)(s) = true$. Hence, $x_j \in \mathcal{J}(\mathcal{L})$ and $x_j \sqsubseteq \bigsqcup\{x_i|\varrho^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\}$ implies $\varrho^{x_j}(R)(s) = true$ where $s \in \mathcal{U}^k, R \in \mathcal{R}$ and $1 \leq j \leq n$. $\square$

Lemma 4.7 The functions $\mathbf{f}$ and $\mathbf{b}$ are monotone, $\mathbf{b} \circ \mathbf{f} = id_{\mathcal{I}}$ and $\mathbf{f} \circ \mathbf{b} = id_{\mathcal{I}^2}$ where $id_{\mathcal{I}}$ and $id_{\mathcal{I}^2}$ are the identity functions over $\mathcal{I}$ and $\mathcal{I}^2$ respectively.

Proof. We first prove that $\mathbf{f}$ is monotone. Assume that $\varrho_1, \varrho_2 \in \mathcal{I}$, $\varrho_1 \sqsubseteq \varrho_2$, $\mathbf{f}(\varrho_1) = (\varrho_1^{x_1},...,\varrho_1^{x_n})$ and $\mathbf{f}(\varrho_2) = (\varrho_2^{x_1},...,\varrho_2^{x_n})$. We want to show that $(\varrho_1^{x_1},...,\varrho_1^{x_n}) \leq^2 (\varrho_2^{x_1},...,\varrho_2^{x_n})$. We prove by contradiction.

Assume that $\exists s \in \mathcal{U}^k, \exists R \in \mathcal{R}, \exists 1 \leq i \leq n : \varrho_2^{x_i}(R)(s) < \varrho_1^{x_i}(R)(s)$. This means $\varrho_2^{x_i}(R)(s) = false$ and $\varrho_1^{x_i}(R)(s) = true$. According to the definition of $\mathbf{f}$ and $\varrho_1^{x_i}(R)(s) = true$, we know that $x_i \sqsubseteq \varrho_1(R)(s)$. From $\varrho_1 \sqsubseteq \varrho_2$, we know that $\varrho_1(R)(s) \sqsubseteq \varrho_2(R)(s)$. Therefore, $x_i \sqsubseteq \varrho_2(R)(s)$. However, from the definition of $\mathbf{f}$ and $\varrho_2^{x_i}(R)(s) = false$, we know that $x_i \not\sqsubseteq \varrho_2(R)(s)$. This is a contradiction.

We now prove that $\mathbf{b}$ is monotone. Assume that $(\varrho_1^{x_1},...,\varrho_1^{x_n}), (\varrho_2^{x_1},...,\varrho_2^{x_n}) \in \mathcal{I}^2$, $(\varrho_1^{x_1},...,\varrho_1^{x_n}) \leq^2 (\varrho_2^{x_1},...,\varrho_2^{x_n})$, $\mathbf{b}((\varrho_1^{x_1},...,\varrho_1^{x_n})) = \varrho_1$ and $\mathbf{b}((\varrho_2^{x_1},...,\varrho_2^{x_n})) = \varrho_2$. We want to show that $\varrho_1 \sqsubseteq \varrho_2$.

Given $s \in \mathcal{U}^k$ and $R \in \mathcal{R}$, according to the definition of $\mathbf{b}$, we know that $\varrho_1(R)(s) = \bigsqcup\{x_i|\varrho_1^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\}$ and $\varrho_2(R)(s) = \bigsqcup\{x_i|\varrho_2^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\}$. Since $(\varrho_1^{x_1},...,\varrho_1^{x_n}) \leq^2 (\varrho_2^{x_1},...,\varrho_2^{x_n})$, we know that $\{x_i|\varrho_1^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\} \subseteq \{x_i|\varrho_2^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\}$. Therefore, $\varrho_1(R)(s) = \bigsqcup\{x_i|\varrho_1^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\} \sqsubseteq \bigsqcup\{x_i|\varrho_2^{x_i}(R)(s) = true \wedge 1 \leq i \leq n\} = \varrho_2(R)(s)$. Since s and R are chosen arbitrarily, we know that $\varrho_1 \sqsubseteq \varrho_2$.

We now show that $b \circ f$ is an identity function over $\mathcal{I}$. Given $s \in \mathcal{U}^k$, $R \in \mathcal{R}$ and

$\varrho \in \mathcal{I}$. Let $\mathbf{f}(\varrho) = (\varrho^{x_1}, ..., \varrho^{x_n})$ and $\mathbf{b}((\varrho^{x_1}, ..., \varrho^{x_n})) = \varrho'$. Therefore, $(b \circ f)(\varrho) = \varrho'$. From the definition of $\mathbf{f}$, we know that $\varrho^{x_i}(R)(s) = true$ iff $x_i \sqsubseteq \varrho(R)(s)$. From Proposition B.3, we know that $\varrho(R)(s) = \bigsqcup\{x | x \in \mathcal{J}(\mathcal{L}) \wedge x \sqsubseteq \varrho(R)(s)\}$. Therefore, we know that $\varrho(R)(s) = \bigsqcup\{x_i | \varrho^{x_i}(R)(s) = true\}$. From the definition of $\mathbf{b}$, we know that $\varrho'(R)(s) = \bigsqcup\{x_i | \varrho^{x_i}(R)(s) = true\}$. Therefore, $\varrho(R)(s) = \varrho'(R)(s)$ Since s and R are chosen arbitrarily, we know that $\varrho = \varrho'$. This means $b \circ f$ is an identity function over $\mathcal{I}$.

We now prove the case of $f \circ b$. Given $s \in \mathcal{U}^k$, $R \in \mathcal{R}$ and $(\varrho^{x_1}, ..., \varrho^{x_n}) \in \mathcal{I}^2$. Let $\mathbf{b}((\varrho^{x_1}, ..., \varrho^{x_n})) = \varrho'$ and $\mathbf{f}(\varrho') = (\varrho'^{x_1}, ..., \varrho'^{x_n})$. Therefore, $(f \circ b)((\varrho^{x_1}, ..., \varrho^{x_n})) = (\varrho'^{x_1}, ..., \varrho'^{x_n})$. From the definition of $\mathbf{b}$, we know that $\varrho'(R)(s) = \bigsqcup\{x_i | \varrho^{x_i}(R)(s) = true\}$. From the definition of $\mathbf{f}$ and Proposition B.3, we know that $\varrho'(R)(s) = \bigsqcup\{x | x \in \mathcal{J}(\mathcal{L}) \wedge x \sqsubseteq \varrho'(R)(s)\} = \bigsqcup\{x_i | \varrho'^{x_i}(R)(s) = true\}$. Therefore, we know that $\bigsqcup\{x_i | \varrho^{x_i}(R)(s) = true\} = \bigsqcup\{x_i | \varrho'^{x_i}(R)(s) = true\}$.

Since s and R are chosen arbitrarily, we can know that $(\varrho^{x_1}, ..., \varrho^{x_n}) = (\varrho'^{x_1}, ..., \varrho'^{x_n})$ iff $\{x_i | \varrho^{x_i}(R)(s) = true\} = \{x_i | \varrho'^{x_i}(R)(s) = true\}$. Assume that $x_i \in \{x_i | \varrho^{x_i}(R)(s) = true\}$. We know that $x_i \sqsubseteq \bigsqcup\{x_i | \varrho'^{x_i}(R)(s) = true\}$. From $\mathcal{C}((\varrho'^{x_1}, ..., \varrho'^{x_n}))$ and Lemma B.5, we know that $\varrho'^{x_i}(R)(s) = true$, which means $x_i \in \{x_i | \varrho'^{x_i}(R)(s) = true\}$. Therefore, $\{x_i | \varrho^{x_i}(R)(s) = true\} \subseteq \{x_i | \varrho'^{x_i}(R)(s) = true\}$. Assume that $x_i \in \{x_i | \varrho'^{x_i}(R)(s) = true\}$. We know that $x_i \sqsubseteq \bigsqcup\{x_i | \varrho^{x_i}(R)(s) = true\}$. From $\mathcal{C}((\varrho^{x_1}, ..., \varrho^{x_n}))$ and Lemma B.5, we know that $\varrho^{x_i}(R)(s) = true$, which means $x_i \in \{x_i | \varrho^{x_i}(R)(s) = true\}$. Therefore, $\{x_i | \varrho^{x_i}(R)(s) = true\} \supseteq \{x_i | \varrho'^{x_i}(R)(s) = true\}$. Therefore, $\{x_i | \varrho^{x_i}(R)(s) = true\} = \{x_i | \varrho'^{x_i}(R)(s) = true\}$. This proves that $(\varrho^{x_1}, ..., \varrho^{x_n}) = (\varrho'^{x_1}, ..., \varrho'^{x_n})$ and therefore $f \circ b$ is an identity function over $\mathcal{I}^2$. □

Lemma B.6 *Let $\mathcal{L}$ be a finite distributive lattice, $\varrho \in \mathcal{I}$, `pre` be a negation-free precondition and $\mathcal{J}(\mathcal{L}) = \{x_1, ..., x_n\}$. We know that $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\texttt{sat}}\ pre]$ iff $[(\varrho^{x_i}, \sigma)\ \underline{\texttt{sat}}^2\ pre] = true$, where $1 \le i \le n$ and $\mathbf{f}(\varrho) = (\varrho^{x_1}, ..., \varrho^{x_n})$.*

Proof. Let $(v_1, ..., v_m) \in \mathcal{U}^k, R \in \mathcal{R}$ and $1 \le i \le n$. We proceed by structure induction on *pre*.

Case $pre = R(v_1, ..., v_m)$: From the definition of $\mathbf{f}$, we know that $\varrho^{x_i}(R)(v_1, ..., v_m) = true$ iff $x_i \sqsubseteq \varrho(R)(v_1, ..., v_m)$. Therefore, we have $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\texttt{sat}}\ R(v_1, ..., v_m)]$ iff $[(\varrho^{x_i}, \sigma)\ \underline{\texttt{sat}}^2\ R(v_1, ..., v_m)]$.

Case $pre = pre_1 \wedge pre_2$: Assume that $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1 \wedge pre_2]$. According to the semantics of multi-valued ALFP, we know that $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1] \sqcap [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_2]$. Therefore, $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1]$ and $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_2]$. According to the induction hypothesis, we know that $[(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}\ pre_1] = true$ and $[(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}\ pre_2] = true$. Therefore, according to the semantics of two-valued and multi-valued ALFP, we know that $[(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}^2\ pre_1 \wedge pre_2] = true$.

Assume that $[(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}^2\ pre_1 \wedge pre_2] = true$. According to the semantics of two-valued ALFP, we know that $[(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}\ pre_1] = true$ and $[(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}\ pre_2] = true$. According to the induction hypothesis, we know that $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1]$ and $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_2]$. Therefore, $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1] \sqcap [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_2]$. According to the semantics of multi-valued ALFP, we know that $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1 \wedge pre_2]$.

Case $pre = pre_1 \vee pre_2$: Assume that $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1 \vee pre_2]$. According to the semantics of multi-valued ALFP, we know that $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1] \sqcup [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_2]$. From Lemma B.4, we know that $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1]$ or $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_2]$. According to the induction hypothesis, we know that $[(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}\ pre_1] = true$ or $[(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}\ pre_2] = true$. Therefore, according to the semantics of two-valued and multi-valued ALFP, we know that $[(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}^2\ pre_1 \vee pre_2] = true$.

Assume that $[(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}^2\ pre_1 \vee pre_2] = true$. According to the semantics of two-valued ALFP, we know that $[(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}\ pre_1] = true$ or $[(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}\ pre_2] = true$. According to the induction hypothesis, we know that $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1]$ or $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_2]$. Therefore, $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1] \sqcup [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_2]$. According to the semantics of multi-valued ALFP, we know that $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre_1 \vee pre_2]$.

Case $pre = \forall x :\ pre'$: Assume that $x_i \sqsubseteq [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ \forall x :\ pre']$. According to the semantics of multi-valued ALFP, we know that $x_i \sqsubseteq \prod_{a \in \mathcal{U}} \{[(\varrho, \sigma[x \mapsto a])\ \underline{\mathtt{sat}}\ pre']\}$. Therefore, $\forall a \in \mathcal{U} : x_i \sqsubseteq [(\varrho, \sigma[x \mapsto a])\ \underline{\mathtt{sat}}\ pre']$. According to the induction hypothesis, we know that $\forall a \in \mathcal{U} : [(\varrho^{x_i}, \sigma[x \mapsto a])\ \underline{\mathtt{sat}}\ pre'] = true$. Therefore, according to the semantics of two-valued and multi-valued ALFP, we know that $[(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}^2\ \forall x :\ pre'] = true$.

Assume that $[(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}^2\ \forall x :\ pre'] = true$. According to the semantics of two-valued ALFP, we know that $\forall a \in \mathcal{U} : [(\varrho^{x_i}, \sigma[x \mapsto a])\ \underline{\mathtt{sat}}\ pre'] = true$. According to the induction hypothesis, we know that $\forall a \in \mathcal{U} : x_i \sqsubseteq [(\varrho, \sigma[x \mapsto$

$a]) \underline{\mathtt{sat}}\ pre']$. Therefore, $x_i \sqsubseteq \bigsqcap_{a\in\mathcal{U}}\{[(\varrho,\sigma[x\mapsto a])\ \underline{\mathtt{sat}}\ pre']\}$. According to the semantics of multi-valued ALFP, we know that $x_i \sqsubseteq [(\varrho,\sigma)\ \underline{\mathtt{sat}}\ \forall x:\ pre']$.

Case $pre = \exists x:\ pre'$: Assume that $x_i \sqsubseteq [(\varrho,\sigma)\ \underline{\mathtt{sat}}\ \exists x:\ pre']$. According to the semantics of multi-valued ALFP, we know that $x_i \sqsubseteq \bigsqcup_{a\in\mathcal{U}}\{[(\varrho,\sigma[x\mapsto a])\ \underline{\mathtt{sat}}\ pre']\}$. From Lemma B.4, we know that $\exists a\in\mathcal{U}: x_i \sqsubseteq [(\varrho,\sigma[x\mapsto a])\ \underline{\mathtt{sat}}\ pre']$. According to the induction hypothesis, we know that $\exists a\in\mathcal{U}: [(\varrho^{x_i},\sigma[x\mapsto a])\ \underline{\mathtt{sat}}\ pre'] = true$. Therefore, according to the semantics of two-valued and multi-valued ALFP, we know that $[(\varrho^{x_i},\sigma)\ \underline{\mathtt{sat}}^2\ \exists x:\ pre'] = true$.

Assume that $[(\varrho^{x_i},\sigma)\ \underline{\mathtt{sat}}^2\ \exists x:\ pre'] = true$. According to the semantics of two-valued ALFP, we know that $\exists a\in\mathcal{U}: [(\varrho^{x_i},\sigma[x\mapsto a])\ \underline{\mathtt{sat}}\ pre'] = true$. According to the induction hypothesis, we know that $\exists a\in\mathcal{U}: x_i \sqsubseteq [(\varrho,\sigma[x\mapsto a])\ \underline{\mathtt{sat}}\ pre']$. Therefore, $x_i \sqsubseteq \bigsqcup_{a\in\mathcal{U}}\{[(\varrho,\sigma[x\mapsto a])\ \underline{\mathtt{sat}}\ pre']\}$. According to the semantics of multi-valued ALFP, we know that $x_i \sqsubseteq [(\varrho,\sigma)\ \underline{\mathtt{sat}}\ \exists x:\ pre']$. □

Lemma B.7 *Let $\mathcal{L}$ be a finite distributive lattice,* $\mathtt{pre}$ *be a negation-free precondition, $\mathcal{J}(\mathcal{L}) = \{x_1,...,x_n\}$ and $(\varrho^{x_1},...,\varrho^{x_n}) \in \mathcal{I}^2$. We have $[(\varrho,\sigma)\ \underline{\mathtt{sat}}\ pre] = \bigsqcup\{x_i | [(\varrho^{x_i},\sigma)\ \underline{\mathtt{sat}}^2\ pre] = true\}$, where $\varrho = \mathbf{b}((\varrho^{x_1},...,\varrho^{x_n}))$.*

Proof. According to Proposition B.3, we know that $[(\varrho,\sigma)\ \underline{\mathtt{sat}}\ pre] = \bigsqcup\{x_i | x_i \sqsubseteq [(\varrho,\sigma)\ \underline{\mathtt{sat}}\ pre]\}$. Since $f\circ b$ is an identity function, we know that $\mathbf{f}(\varrho) = \mathbf{f}(\mathbf{b}((\varrho^{x_1},...,\varrho^{x_n}))) = (\varrho^{x_1},...,\varrho^{x_n})$. According to Lemma B.6, we know that $x_i \sqsubseteq [(\varrho,\sigma)\ \underline{\mathtt{sat}}\ pre]$ iff $[(\varrho^{x_i},\sigma)\ \underline{\mathtt{sat}}^2\ pre] = true$. Therefore, $[(\varrho,\sigma)\ \underline{\mathtt{sat}}\ pre] = \bigsqcup\{x_i | [(\varrho^{x_i},\sigma)\ \underline{\mathtt{sat}}^2\ pre] = true\}$. □

Theorem 4.9 Given ϱ_0 and a negation-free multi-valued ALFP clause cl. The two posets $(\mathcal{I}_{cl,\varrho_0,\sigma_0},\sqsubseteq)$ and $(\mathcal{I}^2_{cl,\varrho_0,\sigma_0},\leq^2)$ are isomorphic.

Proof. It's obvious that $\mathcal{I}_{cl,\varrho_0,\sigma_0} \subseteq \mathcal{I}$ and $\mathcal{I}^2_{cl,\varrho_0,\sigma_0} \subseteq \mathcal{I}^2$. From Corollary 4.8, we know that we only need to show that $\mathbf{f}(\mathcal{I}_{cl,\varrho_0,\sigma_0}) \subseteq \mathcal{I}^2_{cl,\varrho_0,\sigma_0}$ and $\mathbf{b}(\mathcal{I}^2_{cl,\varrho_0,\sigma_0}) \subseteq \mathcal{I}_{cl,\varrho_0,\sigma_0}$.

We first prove that $\mathbf{f}(\mathcal{I}_{cl,\varrho_0,\sigma_0}) \subseteq \mathcal{I}^2_{cl,\varrho_0,\sigma_0}$. Assume that $\varrho \in \mathcal{I}_{cl,\varrho_0,\sigma_0}$ and $\mathbf{f}(\varrho) = (\varrho^{x_1},...,\varrho^{x_n})$. To show $(\varrho^{x_1},...,\varrho^{x_n}) \in \mathcal{I}^2_{cl,\varrho_0,\sigma_0}$, we need to prove that $\forall 1\leq i\leq n: [(\varrho^{x_i},\sigma_0)\ \underline{\mathtt{sat}}^2\ cl] = true \wedge \mathbf{f}(\varrho_0) \leq^2 (\varrho^{x_1},...,\varrho^{x_n}) \wedge \mathcal{C}((\varrho^{x_1},...,\varrho^{x_n}))\}$.

Notice that according to Lemma 4.7, $\mathbf{f}$ is monotone. Therefore, $\mathbf{f}(\varrho_0) \leq^2 \mathbf{f}(\varrho)$, which means $\mathbf{f}(\varrho_0) \leq^2 (\varrho^{x_1}, ..., \varrho^{x_n})$ always holds. Also from the definition of $\mathbf{f}$, we know that $\mathcal{C}((\varrho^{x_1}, ..., \varrho^{x_n}))$ holds. We proceed by structural induction on cl to show that $\forall 1 \leq i \leq n : [(\varrho^{x_i}, \sigma_0) \ \underline{\mathtt{sat}}^2 \ cl] = true$.

Case $cl = \mathbf{true}$: Assume that $\varrho \in \mathcal{I}_{\mathbf{true},\varrho_0,\sigma_0}$. According to the semantics of two-valued ALFP, we know that $\forall 1 \leq i \leq n : [(\varrho^{x_i}, \sigma_0) \ \underline{\mathtt{sat}}^2 \ \mathbf{true}] = true$ holds. Since $\mathbf{f}(\varrho_0) \leq^2 (\varrho^{x_1}, ..., \varrho^{x_n})$ and $\mathcal{C}((\varrho^{x_1}, ..., \varrho^{x_n}))$ always hold, we know that $\mathbf{f}(\varrho) \in \mathcal{I}^2_{\mathbf{true},\varrho_0,\sigma_0}$. Therefore, $\mathbf{f}(\mathcal{I}_{\mathbf{true},\varrho_0,\sigma_0}) \subseteq \mathcal{I}^2_{\mathbf{true},\varrho_0,\sigma_0}$.

Case $cl = cl_1 \wedge cl_2$: According to the semantics of two-valued and multi-valued ALFP, we know that $\mathcal{I}_{cl_1 \wedge cl_2,\varrho_0,\sigma_0} = \mathcal{I}_{cl_1,\varrho_0,\sigma_0} \cap \mathcal{I}_{cl_2,\varrho_0,\sigma_0}$ and $\mathcal{I}^2_{cl_1 \wedge cl_2,\varrho_0,\sigma_0} = \mathcal{I}^2_{cl_1,\varrho_0,\sigma_0} \cap \mathcal{I}^2_{cl_2,\varrho_0,\sigma_0}$. According to the induction hypothesis, we have $\mathbf{f}(\mathcal{I}_{cl_1,\varrho_0,\sigma_0}) \subseteq \mathcal{I}^2_{cl_1,\varrho_0,\sigma_0}$ and $\mathbf{f}(\mathcal{I}_{cl_2,\varrho_0,\sigma_0}) \subseteq \mathcal{I}^2_{cl_2,\varrho_0,\sigma_0}$. Therefore, $\mathbf{f}(\mathcal{I}_{cl_1 \wedge cl_2,\varrho_0,\sigma_0}) = \mathbf{f}(\mathcal{I}_{cl_1,\varrho_0,\sigma_0} \cap \mathcal{I}_{cl_2,\varrho_0,\sigma_0}) \subseteq \mathbf{f}(\mathcal{I}_{cl_1,\varrho_0,\sigma_0}) \cap \mathbf{f}(\mathcal{I}_{cl_2,\varrho_0,\sigma_0}) \subseteq \mathcal{I}^2_{cl_1,\varrho_0,\sigma_0} \cap \mathcal{I}^2_{cl_2,\varrho_0,\sigma_0} = \mathcal{I}^2_{cl_1 \wedge cl_2,\varrho_0,\sigma_0}$.

Case $cl = \forall x : cl'$: According to the semantics of two-valued and multi-valued ALFP, we know that $\mathcal{I}_{\forall x:cl',\varrho_0,\sigma_0} = \bigcap_{a \in \mathcal{U}} \mathcal{I}_{cl',\varrho_0,\sigma_0[x \mapsto a]}$ and $\mathcal{I}^2_{\forall x:cl',\varrho_0,\sigma_0} = \bigcap_{a \in \mathcal{U}} \mathcal{I}^2_{cl',\varrho_0,\sigma_0[x \mapsto a]}$. According to the induction hypothesis, we know that $\forall a \in \mathcal{U} : \mathbf{f}(\mathcal{I}_{cl',\varrho_0,\sigma_0[x \mapsto a]}) \subseteq \mathcal{I}^2_{cl',\varrho_0,\sigma_0[x \mapsto a]}$. Therefore, $\mathbf{f}(\mathcal{I}_{\forall x:cl',\varrho_0,\sigma_0}) = \mathbf{f}(\bigcap_{a \in \mathcal{U}} \mathcal{I}_{cl',\varrho_0,\sigma_0[x \mapsto a]}) \subseteq \bigcap_{a \in \mathcal{U}} \mathbf{f}(\mathcal{I}_{cl',\varrho_0,\sigma_0[x \mapsto a]}) \subseteq \bigcap_{a \in \mathcal{U}} \mathcal{I}^2_{cl',\varrho_0,\sigma_0[x \mapsto a]} = \mathcal{I}^2_{\forall x:cl',\varrho_0,\sigma_0}$.

Case $cl = pre \Rightarrow R(v_1, ..., v_n)$: Assume that $\varrho \in \mathcal{I}_{pre \Rightarrow R(v_1,...,v_n),\varrho_0,\sigma_0}$. According to the semantics of multi-valued ALFP, we know that $[(\varrho, \sigma) \ \underline{\mathtt{sat}} \ pre] \sqsubseteq [(\varrho, \sigma) \ \underline{\mathtt{sat}} \ R(v_1, ..., v_n)]$. Assume that $[(\varrho^{x_i}, \sigma) \ \underline{\mathtt{sat}}^2 \ pre] = true$. From Lemma B.6, we know that $x_i \sqsubseteq [(\varrho, \sigma) \ \underline{\mathtt{sat}} \ pre]$. Therefore, $x_i \sqsubseteq [(\varrho, \sigma) \ \underline{\mathtt{sat}} \ R(v_1, ..., v_n)]$. From the definition of $\mathbf{f}$, we know that $[(\varrho^{x_i}, \sigma) \ \underline{\mathtt{sat}}^2 \ R(v_1, ..., v_n)] = true$. Therefore, according to the semantics of two-valued ALFP, we know that $[(\varrho^{x_i}, \sigma) \ \underline{\mathtt{sat}}^2 \ pre \Rightarrow R(v_1, ..., v_n)] = true$. Hence, $\forall 1 \leq i \leq n : [(\varrho^{x_i}, \sigma) \ \underline{\mathtt{sat}}^2 \ pre \Rightarrow R(v_1, ..., v_n)] = true$. Since $\mathbf{f}(\varrho_0) \leq^2 (\varrho^{x_1}, ..., \varrho^{x_n})$ and $\mathcal{C}((\varrho^{x_1}, ..., \varrho^{x_n}))$ always hold, we know that $\mathbf{f}(\varrho) \in \mathcal{I}^2_{pre \Rightarrow R(v_1,...,v_n),\varrho_0,\sigma_0}$. Therefore, we know that $\mathbf{f}(\mathcal{I}_{pre \Rightarrow R(v_1,...,v_n),\varrho_0,\sigma_0}) \subseteq \mathcal{I}^2_{pre \Rightarrow R(v_1,...,v_n),\varrho_0,\sigma_0}$.

We now show that $\mathbf{b}(\mathcal{I}^2_{cl,\varrho_0,\sigma_0}) \subseteq \mathcal{I}_{cl,\varrho_0,\sigma_0}$. Assume that $(\varrho^{x_1}, ..., \varrho^{x_n}) \in \mathcal{I}^2_{cl,\varrho_0,\sigma_0}$ and $\mathbf{b}((\varrho^{x_1}, ..., \varrho^{x_n})) = \varrho$. Notice that according to Lemma 4.7, $\mathbf{b}$ is monotone and $b \circ f$ is an identity function. Therefore, from $\mathbf{f}(\varrho_0) \leq^2 (\varrho^{x_1}, ..., \varrho^{x_n})$, we know that $\mathbf{b}(\mathbf{f}(\varrho_0)) \leq^2 \mathbf{b}((\varrho^{x_1}, ..., \varrho^{x_n}))$, which means $\varrho_0 \sqsubseteq \varrho$. We proceed by structural induction on cl to show that $[(\varrho, \sigma_0)\ \underline{\mathtt{sat}}\ cl] = true$.

Case cl = **true**: Assume that $(\varrho^{x_1}, ..., \varrho^{x_n}) \in \mathcal{I}^2_{\mathbf{true},\varrho_0,\sigma_0}$. According to the multi-valued semantics of ALFP, we have $[(\varrho, \sigma_0)\ \underline{\mathtt{sat}}\ \mathbf{true}] = true$. Since $\varrho_0 \sqsubseteq \varrho$. Therefore, $\varrho \in \mathcal{I}_{\mathbf{true},\varrho_0,\sigma_0}$ holds. Hence, we have that $\mathbf{b}((\varrho^{x_1}, ..., \varrho^{x_n})) \in \mathcal{I}_{\mathbf{true},\varrho_0,\sigma_0}$Therefore, $\mathbf{b}(\mathcal{I}^2_{\mathbf{true},\varrho_0,\sigma_0}) \subseteq \mathcal{I}_{\mathbf{true},\varrho_0,\sigma_0}$.

Case $cl = cl_1 \wedge cl_2$: According to the semantics of two-valued and multi-valued ALFP, we know that $\mathcal{I}_{cl_1 \wedge cl_2,\varrho_0,\sigma_0} = \mathcal{I}_{cl_1,\varrho_0,\sigma_0} \cap \mathcal{I}_{cl_2,\varrho_0,\sigma_0}$ and $\mathcal{I}^2_{cl_1 \wedge cl_2,\varrho_0,\sigma_0} = \mathcal{I}^2_{cl_1,\varrho_0,\sigma_0} \cap \mathcal{I}^2_{cl_2,\varrho_0,\sigma_0}$. According to the induction hypothesis, we have $\mathbf{b}(\mathcal{I}^2_{cl_1,\varrho_0,\sigma_0}) \subseteq \mathcal{I}_{cl_1,\varrho_0,\sigma_0}$ and $\mathbf{b}(\mathcal{I}^2_{cl_2,\varrho_0,\sigma_0}) \subseteq \mathcal{I}_{cl_2,\varrho_0,\sigma_0}$. Therefore, $\mathbf{b}(\mathcal{I}^2_{cl_1 \wedge cl_2,\varrho_0,\sigma_0}) = \mathbf{b}(\mathcal{I}_{cl_1,\varrho_0,\sigma_0} \cap \mathcal{I}_{cl_2,\varrho_0,\sigma_0}) \subseteq \mathbf{b}(\mathcal{I}^2_{cl_1,\varrho_0,\sigma_0}) \cap \mathbf{b}(\mathcal{I}^2_{cl_2,\varrho_0,\sigma_0}) \subseteq \mathcal{I}^2_{cl_1,\varrho_0,\sigma_0} \cap \mathcal{I}^2_{cl_2,\varrho_0,\sigma_0} = \mathcal{I}_{cl_1 \wedge cl_2,\varrho_0,\sigma_0}$.

Case $cl = \forall x : cl'$: According to the semantics of two-valued and multi-valued ALFP, we know that $\mathcal{I}_{\forall x:cl',\varrho_0,\sigma_0} = \bigcap_{a \in \mathcal{U}} \mathcal{I}_{cl',\varrho_0,\sigma_0[x \mapsto a]}$ and $\mathcal{I}^2_{\forall x:cl',\varrho_0,\sigma_0} = \bigcap_{a \in \mathcal{U}} \mathcal{I}^2_{cl',\varrho_0,\sigma_0[x \mapsto a]}$. According to the induction hypothesis, we know that $\forall a \in \mathcal{U} : \mathbf{b}(\mathcal{I}^2_{cl',\varrho_0,\sigma_0[x \mapsto a]}) \subseteq \mathcal{I}_{cl',\varrho_0,\sigma_0[x \mapsto a]}$. Therefore, we know that $\mathbf{b}(\mathcal{I}^2_{\forall x:cl',\varrho_0,\sigma_0}) = \mathbf{b}(\bigcap_{a \in \mathcal{U}} \mathcal{I}_{cl',\varrho_0,\sigma_0[x \mapsto a]}) \subseteq \bigcap_{a \in \mathcal{U}} \mathbf{b}(\mathcal{I}_{cl',\varrho_0,\sigma_0[x \mapsto a]}) \subseteq \bigcap_{a \in \mathcal{U}} \mathcal{I}_{cl',\varrho_0,\sigma_0[x \mapsto a]} = \mathcal{I}_{\forall x:cl',\varrho_0,\sigma_0}$.

Case $cl = pre \Rightarrow R(v_1, ..., v_n)$: Let's assume that $(\varrho^{x_1}, ..., \varrho^{x_n}) \in \mathcal{I}^2_{pre \Rightarrow R(v_1,...,v_n),\varrho_0,\sigma_0}$. From Lemma B.7, we know that $[(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre] = \bigsqcup\{x_i | [(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}^2\ pre] = true\}$ and $[(\varrho, \sigma)\ \underline{\mathtt{sat}}\ R(v_1, ..., v_n)] = \bigsqcup\{x_i | [(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}^2\ R(v_1, ..., v_n)] = true\}$. Since $\forall 1 \leq i \leq n : [(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}^2 pre \Rightarrow R(v_1, ..., v_n)] = true$, we know that $[(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}^2 pre] = true$ implies $[(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}^2 R(v_1, ..., v_n)] = true$. Therefore, $\{x_i | [(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}^2\ pre] = true\} \subseteq \{x_i | [(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}^2 R(v_1, ..., v_n)] = true\}$. Hence $\bigsqcup\{x_i | [(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}^2 pre] = true\} \sqsubseteq \bigsqcup\{x_i | [(\varrho^{x_i}, \sigma)\ \underline{\mathtt{sat}}^2\ R(v_1, ..., v_n)] = true\}$. Therefore, $[(\varrho, \sigma)\ \underline{\mathtt{sat}}\ pre] \sqsubseteq [(\varrho, \sigma)\ \underline{\mathtt{sat}}\ R(v_1, ..., v_n)]$. According to the semantics of multi-valued ALFP, we know that $\varrho \in \mathcal{I}_{pre \Rightarrow R(v_1,...,v_n),\varrho_0,\sigma_0}$. Hence, $\mathbf{b}(\mathcal{I}^2_{pre \Rightarrow R(v_1,...,v_n),\varrho_0,\sigma_0}) \subseteq \mathcal{I}_{pre \Rightarrow R(v_1,...,v_n),\varrho_0,\sigma_0}$.$\square$

Lemma B.8 *Given a negation-free 2-valued ALFP clause cl. Assume that* $\varrho_0^1 \leq^2 \varrho_0^2$. *Let* $\varrho_1 = \wedge^2\{\varrho | [(\varrho, \sigma_0)\ \underline{\mathtt{sat}}^2\ cl] = true \wedge \varrho_0^1 \leq^2 \varrho\}$ *and* $\varrho_2 = \wedge^2\{\varrho | [(\varrho, \sigma_0)\ \underline{\mathtt{sat}}^2\ cl] = true \wedge \varrho_0^2 \leq^2 \varrho\}$. *We know that* $\varrho_1 \leq^2 \varrho_2$.

Proof. From Proposition 2.6, we know that $[(\varrho_2,\sigma_0)\ \underline{\mathtt{sat}}^2\ cl] = true \wedge \varrho_0^2 \leq^2 \varrho_2$. From the assumption, we also know that $\varrho_0^1 \leq^2 \varrho_2$. Therefore, ϱ_2 is an element of the set $\{\varrho | [(\varrho,\sigma_0)\ \underline{\mathtt{sat}}^2\ cl] = true \wedge \varrho_0^1 \leq^2 \varrho\}$. Since ϱ_1 is a lower bound of this set, we have $\varrho_1 \leq^2 \varrho_2$. □

Lemma 4.10 Let $\mathcal{M} = (\mathcal{L}, \sim)$ be a finite distributive multi-valued structure. Then $\wedge^2 \mathcal{I}^2_{cl,\varrho_0,\sigma_0} = \wedge^2\{(\varrho^{x_1},...,\varrho^{x_n}) | \forall 1 \leq i \leq n : [(\varrho^{x_i},\sigma_0)\ \underline{\mathtt{sat}}^2\ cl] = true \wedge \mathbf{f}(\varrho_0) \leq^2 (\varrho^{x_1},...,\varrho^{x_n})\}$.

Proof. Let $(\varrho^{x_1},...,\varrho^{x_n}) = \wedge^2\{(\varrho^{x_1},...,\varrho^{x_n}) | \forall 1 \leq i \leq n : [(\varrho^{x_i},\sigma_0)\ \underline{\mathtt{sat}}^2\ cl] = true \wedge \mathbf{f}(\varrho_0) \leq^2 (\varrho^{x_1},...,\varrho^{x_n})\}$. From the definition of $\mathbf{f}$, we know that $x_i \sqsupseteq x_j$ implies $\varrho_0^{x_i} \sqsubseteq \varrho_0^{x_j}$. From Lemma B.8, we know that $\varrho_0^{x_i} \sqsubseteq \varrho_0^{x_j}$ implies $\varrho^{x_i} \sqsubseteq \varrho^{x_j}$. Therefore, $x_i \sqsupseteq x_j$ implies $\varrho^{x_i} \sqsubseteq \varrho^{x_j}$. Therefore, $\mathcal{C}((\varrho^{x_1},...,\varrho^{x_n}))$ holds.

From Proposition 2.6 and above, we know that $(\varrho^{x_1},...,\varrho^{x_n}) \in \mathcal{I}^2_{cl,\varrho_0,\sigma_0}$. Therefore, $\wedge^2 \mathcal{I}^2_{cl,\varrho_0,\sigma_0} \leq^2 (\varrho^{x_1},...,\varrho^{x_n})$. Since it's clear that $\mathcal{I}^2_{cl,\varrho_0,\sigma_0} \subseteq \{(\varrho^{x_1},...,\varrho^{x_n}) | \forall 1 \leq i \leq n : [(\varrho^{x_i},\sigma_0)\ \underline{\mathtt{sat}}^2\ cl] = true \wedge \mathbf{f}(\varrho_0) \leq^2 (\varrho^{x_1},...,\varrho^{x_n})\}$, we know that $(\varrho^{x_1},...,\varrho^{x_n}) \leq^2 \wedge^2 \mathcal{I}^2_{cl,\varrho_0,\sigma_0}$. Since $\leq^2$ is anti-symmetric, we have $\wedge^2 \mathcal{I}^2_{cl,\varrho_0,\sigma_0} = (\varrho^{x_1},...,\varrho^{x_n})$. □

Theorem 4.16 For a CTL formula ϕ and the least model ϱ of $\vec{R} \vdash \phi$ such that $\varrho = \wedge^3_\sharp\{\varrho \mid [(\varrho,\sigma)\ \underline{\mathtt{sat}}^3\ (\vec{R} \vdash \phi)] = true,\ \varrho_0 \leq^3 \varrho\}$, where ϱ_0 defines P_p, T and $True$, we know that $[(M,s) \models^3 \phi] = \varrho(R_\phi)(s)$.

Proof. We actually only have to prove the following two statements: $[(M,s) \models^3 \phi] = true$ iff $\varrho(R_\phi)(s) = true$ and $[(M,s) \models^3 \phi] \geq^3 \perp$ iff $\varrho(R_\phi)(s) \geq^3 \perp$. This is because if the above two statements hold, it's obvious that the following two statements: $[(M,s) \models^3 \phi] = \perp$ iff $\varrho(R_\phi)(s) = \perp$ and $[(M,s) \models^3 \phi] = false$ iff $\varrho(R_\phi)(s) = false$ also hold. Then, the statement $[(M,s) \models^3 \phi] = \varrho(R_\phi)(s)$ also holds. We proceed by structural induction on ϕ. For simplicity, when we say that ϱ is the least model of $\vec{R} \vdash \phi$ in the following, we mean that $\varrho = \wedge^3_\sharp\{\varrho \mid [(\varrho,\sigma)\ \underline{\mathtt{sat}}^3\ (\vec{R} \vdash \phi)] = true,\ \varrho_0 \leq^3 \varrho\}$.

Case $\phi = true$: We have $\{s | [(M,s) \models^3 \mathbf{true}] = true\} = S$ and $\{s | \varrho(R_{true})(s) = true\} = S$. Therefore, we know that $[(M,s) \models^3 \mathbf{true}] = \varrho(R_{true})(s)$.

Case $\phi = p$: We have $\{s|[(M,s) \models^3 p] = true\} = \{s|L(s,p) = true\}$ and $\{s|\varrho(R_p)(s) = true\} = \{s|\varrho(P_p)(s) = true\} = \{s|L(s,p) = true\}$. Therefore, we know that $[(M,s) \models^3 p] = true$ iff $\varrho(R_p)(s) = true$.

We also have $\{s|[(M,s) \models^3 p] \geq^3 \perp\} = \{s|L(s,p) \geq^3 \perp\}$ and $\{s|\varrho(R_p)(s) \geq^3 \perp\} = \{s|\varrho(P_p)(s) \geq^3 \perp\} = \{s|L(s,p) \geq^3 \perp\}$. Therefore, we know that $[(M,s) \models^3 p] \geq^3 \perp$ iff $\varrho(R_p)(s) \geq^3 \perp$.

Case $\phi = \neg\phi'$: Let's consider the least model ϱ for $\vec{R} \vdash \neg\phi'$. Notice that $\varrho(R_{\phi'})$ coincides with $\varrho'(R_{\phi'})$ in the least model ϱ' for $\vec{R} \vdash \phi'$.

According to the semantics of 3-valued CTL, we have $\{s|[(M,s) \models^3 \neg\phi'] = true\} = \{s|\neg^3[(M,s) \models^3 \phi'] = true\} = \{s|[(M,s) \models^3 \phi'] = false\}$. According to the induction hypothesis and 3-valued semantics of ALFP, we have $\{s|\varrho(R_{\neg\phi'})(s) = true\} = \{s|\neg^3\varrho(R_{\phi'})(s) = true\} = \{s|\varrho(R_{\phi'})(s) = false\} = \{s|[(M,s) \models^3 \phi'] = false\}$. Therefore, we know that $[(M,s) \models^3 \neg\phi'] = true$ iff $\varrho(R_{\neg\phi'})(s) = true$.

According to the semantics of 3-valued CTL, we have $\{s|[(M,s) \models^3 \neg\phi'] \geq^3 \perp\} = \{s|\neg^3[(M,s) \models^3 \phi'] \geq^3 \perp\} = \{s|[(M,s) \models^3 \phi'] \leq^3 \perp\}$. According to the induction hypothesis and 3-valued semantics of ALFP, we also have $\{s|\varrho(R_{\neg\phi'})(s) \geq^3 \perp\} = \{s|\neg^3\varrho(R_{\phi'})(s) \geq^3 \perp\} = \{s|\varrho(R_{\phi'})(s) \leq^3 \perp\}$ $= \{s|[(M,s) \models^3 \phi'] \leq^3 \perp\}$. Therefore, we know that $[(M,s) \models^3 \neg\phi'] \geq^3 \perp$ iff $\varrho(R_{\neg\phi'})(s) \geq^3 \perp$.

Case $\phi = \phi_1 \vee \phi_2$: Let's consider the least model ϱ for $\vec{R} \vdash \phi$. In this case, it is possible that ϕ_1 and ϕ_2 have a same subformula. We claim that clauses generated for a same judgement are the same. Therefore, the same subformula in ϕ_1 and ϕ_2 are dealt with in the same way by the flow logic. In the clauses for $\vec{R} \vdash \phi$, we only keep one copy of the clauses for same subformulae in ϕ_1 and ϕ_2. Also notice that $\varrho(R_{\phi_1})$ (or $\varrho(R_{\phi_2})$) coincides with $\varrho'(R_{\phi_1})$ (or $\varrho'(R_{\phi_2})$) in the least model ϱ' of $\vec{R} \vdash \phi_1$ (or $\vec{R} \vdash \phi_2$).

According to the semantics of 3-valued CTL, we have $\{s|[(M,s) \models^3 \phi_1 \vee \phi_2] = true\} = \{s|[(M,s) \models^3 \phi_1] \vee^3 [(M,s) \models^3 \phi_2] = true\} = \{s|[(M,s) \models^3 \phi_1] = true$ or $[(M,s) \models^3 \phi_2] = true\}$. According to the induction hypothesis and 3-valued semantics of ALFP, we also have $\{s|\varrho(R_{\phi_1 \vee \phi_2})(s) = true\} = \{s|\varrho(R_{\phi_1})(s) \vee^3$

$\varrho(R_{\phi_2})(s) = true\} = \{s|\varrho(R_{\phi_1})(s) = true$ or $\varrho(R_{\phi_2})(s) = true\} = \{s|[(M,s) \models^3 \phi_1] = true$ or $[(M,s) \models^3 \phi_2] = true\}$. Therefore, we know that $[(M,s) \models^3 \phi_1 \vee \phi_2] = true$ iff $\varrho(R_{\phi_1 \vee \phi_2})(s) = true$.

According to the semantics of 3-valued CTL, we have $\{s|[(M,s) \models^3 \phi_1 \vee \phi_2] \geq^3 \bot\} = \{s|[(M,s) \models^3 \phi_1] \vee^3 [(M,s) \models^3 \phi_2] \geq^3 \bot\} = \{s|[(M,s) \models^3 \phi_1] \geq^3 \bot$ or $[(M,s) \models^3 \phi_2] \geq^3 \bot\}$. According to the induction hypothesis and 3-valued semantics of ALFP, we also have $\{s|\varrho(R_{\phi_1 \vee \phi_2})(s) \geq^3 \bot\} = \{s|\varrho(R_{\phi_1})(s) \vee^3 \varrho(R_{\phi_2})(s) \geq^3 \bot\} = \{s|\varrho(R_{\phi_1})(s) \geq^3 \bot$ or $\varrho(R_{\phi_2})(s) \geq^3 \bot\} = \{s|[(M,s) \models^3 \phi_1] \geq^3 \bot$ or $[(M,s) \models^3 \phi_2] \geq^3 \bot\}$. Therefore, we know that $[(M,s) \models^3 \phi_1 \vee \phi_2] \geq^3 \bot$ iff $\varrho(R_{\phi_1 \vee \phi_2})(s) \geq^3 \bot$.

Case $\phi = \mathbf{EX}\phi'$: Let's consider the least model ϱ for $\vec{R} \vdash \mathbf{EX}\phi'$. Notice that $\varrho(R_{\phi'})$ coincides with $\varrho'(R_{\phi'})$ in the least model ϱ' of $\vec{R} \vdash \phi'$.

According to the semantics of 3-valued CTL, we have $\{s|[(M,s) \models^3 \mathbf{EX}\phi'] = true\} = \{s|$ there exists a must path π from $s : |\pi| > 1 \wedge [(M,\pi[1]) \models^3 \phi'] = true\}$. According to the assumptions and the induction hypothesis and 3-valued semantics of ALFP, we have $\{s|\varrho(R_{\mathbf{EX}\phi'})(s) = true\} = \{s|\exists s' : \varrho(T)(s,s') \wedge^3 \varrho(R_{\phi'})(s') = true\} = \{s|\exists s' : \varrho(T)(s,s') = true$ and $\varrho(R_{\phi'})(s') = true\} = \{s|\exists s' : s \xrightarrow{must} s'$ such that $[(M,s') \models^3 \phi'] = true\}$. We now begin to prove that $\{s|\varrho(R_{\mathbf{EX}\phi'})(s) = true\} = \{s|\ [(M,s) \models^3 \mathbf{EX}\phi'] = true\}$.

It's obvious that $\{s|\ [(M,s) \models^3 \mathbf{EX}\phi'] = true\} \subseteq \{s|\varrho(R_{\mathbf{EX}\phi'})(s) = true\}$. Assume that we have a must transition $s \xrightarrow{must} s'$ such that $[(M,s') \models^3 \phi'] = true$. We can extend $s \xrightarrow{must} s'$ to a must path π such that $\pi[0] = s$ and $\pi[1] = s'$. This proves the other inclusion $\{s|\varrho(R_{\mathbf{EX}\phi'})(s) = true\} \subseteq \{s|\ [(M,s) \models^3 \mathbf{EX}\phi'] = true\}$. Therefore, we have $[(M,s) \models^3 \mathbf{EX}\phi'] = true$ iff $\varrho(R_{\mathbf{EX}\phi'})(s) = true$.

According to the semantics of 3-valued CTL, we have $\{s|[(M,s) \models^3 \mathbf{EX}\phi'] \geq^3 \bot\} = \{s|$ there exists a may path π from $s : [(M,\pi[1]) \models^3 \phi'] \neq false\} = \{s|$ there exists a may path π from $s : [(M,\pi[1]) \models^3 \phi'] \geq^3 \bot\}$. According to the assumptions and the induction hypothesis and 3-valued semantics of ALFP, we have $\{s|\varrho(R_{\mathbf{EX}\phi'})(s) \geq^3 \bot\} = \{s|\exists s' : \varrho(T)(s,s') \wedge^3 \varrho(R_{\phi'})(s') \geq^3 \bot\} = \{s|\exists s' : \varrho(T)(s,s') \geq^3 \bot$ and $\varrho(R_{\phi'})(s') \geq^3 \bot\} = \{s|\exists s' : s \xrightarrow{may} s'$ such that $[(M,s') \models^3 \phi'] \geq^3 \bot\}$. We now begin to prove that $\{s|\varrho(R_{\mathbf{EX}\phi'})(s) \geq^3 \bot\} = \{s|\ [(M,s) \models^3 \mathbf{EX}\phi'] \geq^3 \bot\}$.

It's obvious that $\{s|\ [(M,s) \models^3 \mathbf{EX}\phi'] \geq^3 \bot\} \subseteq \{s|\varrho(R_{\mathbf{EX}\phi'})(s) \geq^3 \bot\}$. Assume that we have a may transition $s \overset{may}{\longrightarrow} s'$ such that $[(M,s') \models^3 \phi'] \geq^3 \bot$. We can extend $s \overset{may}{\longrightarrow} s'$ to a may path π such that $\pi[0] = s$ and $\pi[1] = s'$. This proves the other inclusion $\{s|\varrho(R_{\mathbf{EX}\phi'})(s) \geq^3 \bot\} \subseteq \{s|\ [(M,s) \models^3 \mathbf{EX}\phi'] \geq^3 \bot\}$. Therefore, we have $[(M,s) \models^3 \mathbf{EX}\phi'] \geq^3 \bot$ iff $\varrho(R_{\mathbf{EX}\phi'})(s) \geq^3 \bot$.

Case $\phi = \mathbf{E}[\phi_1 \mathbf{U} \phi_2]$: Let's consider the least model for $\vec{R} \vdash \mathbf{E}[\phi_1 \mathbf{U} \phi_2]$. In this case, it is also possible that ϕ_1 and ϕ_2 have a same subformula. Similarly, we generate same clauses for the same judgement. Therefore, the same subformula in ϕ_1 and ϕ_2 are dealt with in the same way by flow logic. In the clauses for $\mathbf{E}[\phi_1 \mathbf{U} \phi_2]$, we only keep one copy of the clauses for same subformulae in ϕ_1 and ϕ_2. Also notice that $\varrho(R_{\phi_1})$ (or $\varrho(R_{\phi_2})$) coincides with $\varrho'(R_{\phi_1})$ (or $\varrho'(R_{\phi_2})$) in the least model ϱ' of $\vec{R} \vdash \phi_1$ (or $\vec{R} \vdash \phi_2$).

According to the semantics of 3-valued CTL, we have $\{s|[(M,s) \models^3 \mathbf{E}[\phi_1 \mathbf{U} \phi_2]] = true\} = \bigcup_K S^K (K > 0)$, where $S^K = \{s|$ there exists a must path π from $s : |\pi| \geq K \wedge [(M,\pi[K]) \models^3 \phi_2] = true$ and $\forall\ 0 \leq j < K : [(M,\pi[j]) \models^3 \phi_1] = true\}$. According to the assumptions and the induction hypothesis and 3-valued semantics of ALFP, we have $\{s|\varrho(R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]})(s) = true\} = \bigcup_K R^K (K > 0)$, where $R^0 = \{s|\varrho(R_{\phi_2})(s) = true\} = \{s|[(M,s) \models^3 \phi_2] = true\}$ and $R^K = \{s|\exists s' : \varrho(T)(s,s') = true \wedge \varrho(R_{\phi_1})(s) = true \wedge s' \in R^{K-1}\} = \{s|\exists s' : s \overset{must}{\longrightarrow} s' \wedge [(M,s) \models^3 \phi_1] = true \wedge s' \in R^{K-1}\}$.

We prove $R^K = S^K$ by induction on K.

The base case is when $K = 0$. It is obvious that $S^0 \subseteq R^0$. Assume that for state s, we have $[(M,s) \models^3 \phi_2] = true$. We can extend s to a must path π from s and obviously we have $|\pi| \geq 0$ and $[(M,\pi[0]) \models^3 \phi_2] = true$. This proves $R^0 \subseteq S^0$.

Let's consider $K+1$. $R^{K+1} = \{s|\exists s' : s \overset{must}{\longrightarrow} s' \wedge [(M,s) \models^3 \phi_1] = true \wedge s' \in R^K\}$. According to the induction hypothesis, we know that $R^{K+1} = \{s|\exists s' : s \overset{must}{\longrightarrow} s' \wedge [(M,s) \models^3 \phi_1] = true \wedge s' \in S^K\} = \{s|\exists s' : s \overset{must}{\longrightarrow} s' \wedge [(M,s) \models^3 \phi_1] = true$ and there exists a must path π from s' such that $|\pi| \geq K \wedge [(M,\pi[K]) \models^3 \phi_2] = true \wedge \forall\ 0 \leq j < K : [(M,\pi[j]) \models^3 \phi_1] = true\}$. Assume that $s \in R^{K+1}$, we can extend the transition $s \overset{must}{\longrightarrow} s'$ to a path π' by appending the path π starting from s' to $s \overset{must}{\longrightarrow} s'$ such that $\pi'[0] = s$ and $\pi'[k+1] = \pi[k] (0 \leq k \leq K)$. Now we know that there exists a must path π'

from s such that $|\pi'| \geq K+1 \wedge [(M, \pi'[K+1]) \models^3 \phi_2] = true \wedge \forall\ 0 \leq j < K+1 :$ $[(M, \pi'[j]) \models^3 \phi_1] = true$. Therefore, $s \in S^{K+1}$. This proves $R^{K+1} \subseteq S^{K+1}$.

For the other direction, assume that $s \in S^{K+1}$. Then there exists a must path π from s such that $|\pi| \geq K+1 \wedge [(M, \pi[K+1]) \models^3 \phi_2] = true \wedge \forall\ 0 \leq j < K+1 : [(M, \pi[j]) \models^3 \phi_1] = true$. Consider the suffix π' of the path π such that $\pi'[k] = \pi[k+1](0 \leq k \leq K)$. It's obvious that π' is a must path and $|\pi| \geq K \wedge [(M, \pi'[K]) \models^3 \phi_2] = true \wedge \forall\ 0 \leq j < K : [(M, \pi'[j]) \models^3 \phi_1] = true$. This means $\pi'[0] \in S^K$ and according to the induction hypothesis we have $\pi'[0] \in R^K$. Therefore, we know that there exists $s'(s' = \pi'[0])$ such that $s \xrightarrow{must} s' \wedge [(M, s) \models^3 \phi_1] = true \wedge s' \in R^K$. This means $s \in R^{K+1}$. Therefore, we have $S^{K+1} \subseteq R^{K+1}$.

From above, we have $[(M, s) \models^3 \mathbf{E}[\phi_1 \mathbf{U} \phi_2]] = true$ iff $\varrho(R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]})(s) = true$.

According to the semantics of 3-valued CTL, we have $\{s | [(M, s) \models^3 \mathbf{E}[\phi_1 \mathbf{U} \phi_2]] \geq^3 \perp\} = \bigcup_K S^K_{tt|\perp}(K > 0)$, where $S^K_{tt|\perp} = \{s|$ there exists a may path π from $s : [(M, \pi[K]) \models^3 \phi_2] \neq false \wedge \forall\ 0 \leq j < K : [(M, \pi[j]) \models^3 \phi_1] \neq false\} = \{s|$ there exists a may path π from $s : [(M, \pi[K]) \models^3 \phi_2] \geq^3 \perp \wedge \forall\ 0 \leq j < K :$ $[(M, \pi[j]) \models^3 \phi_1] \geq^3 \perp\}$. According to the assumptions and the induction hypothesis and 3-valued semantics of ALFP, we have $\{s | \varrho(R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]})(s) \geq^3 \perp\} = \bigcup_K R^K_{tt|\perp}(K > 0)$, where $R^0_{tt|\perp} = \{s | \varrho(R_{\phi_2})(s) \geq^3 \perp\} = \{s | [(M, s) \models^3 \phi_2] \geq^3 \perp\}$ and $R^K_{tt|\perp} = \{s | \exists s' : \varrho(T)(s, s') \geq^3 \perp \wedge \varrho(R_{\phi_1})(s) \geq^3 \perp \wedge s' \in R^{K-1}_{tt|\perp}\} = \{s | \exists s' : s \xrightarrow{may} s' \wedge [(M, s) \models^3 \phi_1] \geq^3 \perp \wedge s' \in R^{K-1}_{tt|\perp}\}$.

We prove $R^K_{tt|\perp} = S^K_{tt|\perp}$ by induction on K.

The base case is when $K = 0$. It is obvious that $S^0_{tt|\perp} \subseteq R^0_{tt|\perp}$. Assume that for state s, we have $[(M, s) \models^3 \phi_2] \geq^3 \perp$. We can extend s to a may path π from s and obviously we have $[(M, \pi[0]) \models^3 \phi_2] \geq^3 \perp$. This proves $R^0_{tt|\perp} \subseteq S^0_{tt|\perp}$.

Let's consider $K+1$. $R^{K+1} = \{s | \exists s' : s \xrightarrow{may} s' \wedge [(M, s) \models^3 \phi_1] \geq^3 \perp \wedge s' \in R^K_{tt|\perp}\}$. According to the induction hypothesis, we know that $R^{K+1} = \{s | \exists s' :$ $s \xrightarrow{may} s' \wedge [(M, s) \models^3 \phi_1] \geq^3 \perp \wedge s' \in S^K_{tt|\perp}\} = \{s | \exists s' : s \xrightarrow{may} s' \wedge [(M, s) \models^3$ $\phi_1] = true$ and there exists a may path π from s' such that $[(M, \pi[K]) \models^3$ $\phi_2] \geq^3 \perp \wedge \forall\ 0 \leq j < K : [(M, \pi[j]) \models^3 \phi_1] \geq^3 \perp\}$. Assume that $s \in R^{K+1}_{tt|\perp}$, we can extend the transition $s \overset{may}{\longrightarrow} s'$ to a path π' by appending the path π starting from s' to $s \overset{may}{\longrightarrow} s'$ such that $\pi'[0] = s$ and $\pi'[k+1] = \pi[k](0 \leq k \leq K)$. Now we know that there exists a may path π' from s such that $[(M, \pi'[K+1]) \models^3 \phi_2] \geq^3 \perp \wedge \forall\ 0 \leq j < K+1 : [(M, \pi'[j]) \models^3 \phi_1] \geq^3 \perp$. Therefore, $s \in S^{K+1}_{tt|\perp}$. This proves $R^{K+1}_{tt|\perp} \subseteq S^{K+1}_{tt|\perp}$.

For the other direction, assume that $s \in S^{K+1}_{tt|\perp}$. Then there exists a may path π from s such that $[(M, \pi[K+1]) \models^3 \phi_2] \geq^3 \perp \wedge \forall\ 0 \leq j < K+1 : [(M, \pi[j]) \models^3 \phi_1] \geq^3 \perp$. Consider the suffix π' of the path π such that $\pi'[k] = \pi[k+1](0 \leq k \leq K)$. It's obvious that π' is a may path and $[(M, \pi'[K]) \models^3 \phi_2] \geq^3 \perp \wedge \forall\ 0 \leq j < K : [(M, \pi'[j]) \models^3 \phi_1] \geq^3 \perp$. This means $\pi'[0] \in S^{K}_{tt|\perp}$ and according to the induction hypothesis we have $\pi'[0] \in R^{K}_{tt|\perp}$. Therefore, we know that there exists $s'(s' = \pi'[0])$ such that $s \overset{may}{\longrightarrow} s' \wedge [(M, s) \models^3 \phi_1] \geq^3 \perp \wedge s' \in R^{K}_{tt|\perp}$. This means $s \in R^{K+1}_{tt|\perp}$. Therefore, we have $S^{K+1}_{tt|\perp} \subseteq R^{K+1}_{tt|\perp}$.

From above, we have $[(M, s) \models^3 \mathbf{E}[\phi_1 \mathbf{U} \phi_2]] \geq^3 \perp$ iff $\varrho(R_{\mathbf{E}[\phi_1 \mathbf{U} \phi_2]})(s) \geq^3 \perp$.

Case $\phi = \mathbf{AF}\phi'$: Let's consider the least model for $\vec{R} \vdash \mathbf{AF}\phi'$. Notice that $\varrho(R_{\phi'})$ coincides with $\varrho'(R_{\phi'})$ in the least model ϱ' of $\vec{R} \vdash \phi'$.

According to the assumptions and the induction hypothesis and 3-valued semantics of ALFP, we have $\{s | \varrho(R_{\mathbf{AF}\phi'})(s) = true\} = \bigcup_K R^K$, where $R^0 = \{s | \varrho(R_{\phi'})(s) = true\} = \{s | [(M, s) \models^3 \phi'] = true\}$ and $R^{K+1} = \{s | \forall s' : \varrho(T)(s, s') = false \vee s' \in \bigcup_{k \leq K} R^k\} \cup R^0 (K \geq 0)$.

The rest of the proof goes in two steps. We first prove that $\{s | [(M, s) \models^3 \mathbf{AF}\phi'] = true\} = \bigcup_K S^K$, where $S^K = \{s |$ for all may path π from s: $\exists k : 0 \leq k \leq K$ such that $[(M, s_k) \models^3 \phi'] = true\}$. Then we will prove by induction on K that $R^K = S^K$.

Now let's proof the first step, that is $\{s | [(M, s) \models^3 \mathbf{AF}\phi'] = true\} = \bigcup_K S^K$. Let's consider the set $T = \{s | [(M, s) \models^3 \mathbf{AF}\phi'] = true\} \setminus \bigcup_K S^K$ and we shall prove that it is empty. We proceed by contradiction. Suppose $T \neq \varnothing$ and choose $s_0 \in T$. It is obvious that $[(M, s_0) \models^3 \mathbf{AF}\phi'] = true$ but $[(M, s_0) \models^3 \phi'] \neq true$ since otherwise $s_0 \in S^0$(contradicting $s_0 \notin \bigcup_K S^K$).

The transition system we consider here is finitely branching, and now we claim that for all *may successors* s_1 of $s_0(s_0 \xrightarrow{may} s_1)$, we have $[(M,s_1) \models^3 \mathbf{AF}\phi'] = true$. Suppose for one may successor s_1 of s_0 we have $[(M,s_1) \models^3 \mathbf{AF}\phi'] \neq true$. Then there exists an infinite may path starting from s_1 ($s_1 \xrightarrow{may} s_2 \xrightarrow{may} ...$) such that for all states along the path, we have $[(M,s_i) \models^3 \phi'] \neq true (i \geq 1)$. Combining $s_0 \xrightarrow{may} s_1$ with the infinite may path $s_1 \xrightarrow{may} s_2 \xrightarrow{may} ...$, we get a new infinite may path $s_0 \xrightarrow{may} s_1 \xrightarrow{may} s_2 \xrightarrow{may} ...$ such that for all states along the new path, we have $[(M,s_i) \models^3 \phi'] \neq true(i \geq 0)$. This means $[(M,s_0) \models^3 \mathbf{AF}\phi'] \neq true$ and contradicts the fact that $s_0 \in T$.

On the other hand, it can't be the case that for all may successors s_1 of s_0, we have $[(M,s_1) \models^3 \phi'] = true$ since otherwise $s_0 \in S^1$ (contradicting $s_0 \notin \bigcup_K S^K$).

We now choose one may successor s_1 of s_0 such that $[(M,s_1) \models^3 \mathbf{AF}\phi'] = true$ but $[(M,s_1) \models^3 \phi'] \neq true$. Similarly, we can also show that for all may successors s_2 of s_1, we have $[(M,s_2) \models^3 \mathbf{AF}\phi'] = true$. It can't be the case that for all may successors s_2 of s_1, we have $[(M,s_2) \models^3 \phi'] = true$ since otherwise $s_0 \in S^2$ (contradicting $s_0 \notin \bigcup_K S^K$). We can choose one may successor s_2 of s_1 such that $[(M,s_2) \models^3 \mathbf{AF}\phi'] = true$ but $[(M,s_2) \models^3 \phi'] \neq true$. This process can continue arbitrarily often and produce an infinite may path starting from s_0 ($s_0 \xrightarrow{may} s_1 \xrightarrow{may} s_2 \xrightarrow{may} ...$) such that for all the states along the path, we have $[(M,s_i) \models^3 \phi'] \neq true(i \geq 0)$. This contradicts the assumption $[(M,s_0) \models^3 \mathbf{AF}\phi'] = true$. Hence $T = \varnothing$.

For the second step, we prove $R^K = S^K$ by induction on K.

When $K = 0$, obviously $R^0 = S^0$.

Let's consider $K+1$. $R^{K+1} = \{s | \forall s' : \varrho(T)(s,s') = false \vee s' \in \bigcup_{k \leq K} R^k\} \cup R^0 (K \geq 0)$. According to the induction hypothesis, $R^{K+1} = \{s | \forall s' : \varrho(T)(s,s') = false \vee s' \in \bigcup_{k \leq K} S^k\} \cup S^0 (K \geq 0)$. It's obvious that $S^k \subseteq S^{k+1} (0 \leq k)$. Therefore, $S^K = \bigcup_{k \leq K} S^k (K \geq 0)$. Then we have $R^{K+1} = \{s | \forall s' : \varrho(T)(s,s') = false \vee s' \in S^K\} \cup S^0 = \{s | \forall s'$: either $\varrho(T)(s,s') = false$ or for all may path π from s': $\exists k : 0 \leq k \leq K$ such that $[(M,\pi[k]) \models^3 \phi'] = true\} \cup \{s | [(M,s) \models^3 \phi'] = true\}$. According to the assumptions, we know that $\varrho(T)(s,s') \neq false$ iff $s \xrightarrow{may} s'$. Assume that $s \in R^{K+1}$, either we know that for all may successors s' of s ($s \stackrel{may}{\longrightarrow} s'$) and for all may path π from s': $\exists k : 0 \leq k \leq K$ such that $[(M, \pi[k]) \models^3 \phi'] = true$, or $[(M, s) \models^3 \phi'] = true$, or both. We have two cases. The first case is when $[(M, s) \models^3 \phi'] = true$. Obviously we have $s \in S^{K+1}$. The second case is when $[(M, s) \models^3 \phi'] \neq true$. In this case, for all may successors s' of s and for all may path π from s': $\exists k : 0 \leq k \leq K$ such that $[(M, \pi[k]) \models^3 \phi'] = true$. We can extend the transition $s \stackrel{may}{\longrightarrow} s'$ to a path π' by appending the path π starting from s' to $s \stackrel{may}{\longrightarrow} s'$ such that $\pi'[0] = s$ and $\pi'[k+1] = \pi[k](0 \leq k \leq K)$. Therefore, we know that for all may path π' from s: $\exists k : 0 \leq k \leq K+1$ such that $[(M, \pi'[k]) \models^3 \phi'] = true$. Therefore, $s \in S^{K+1}$ as well. This proves $R^{K+1} \subseteq S^{K+1}$.

For the other direction, assume that $s \in S^{K+1}$. Then for all may path π from s there exists a number $k(0 \leq k \leq K+1)$ such that $[(M, \pi[k]) \models^3 \phi'] = true$. We have two cases. The first case is when $[(M, s) \models^3 \phi'] = true$. Obviously we have $s \in R^0$ and therefore $s \in R^{K+1}$. The second case is when $[(M, s) \models^3 \phi'] \neq true$. Consider the suffix π' of the path π such that $\pi'[k] = \pi[k+1](0 \leq k \leq K)$. It's obvious that π' is a may path, and there exists a number $k(0 \leq k \leq K)$ such that $[(M, \pi'[k]) \models^3 \phi'] = true$. This means $\pi'[0] \in S^K$. Therefore, $\pi'[0] \in \bigcup_{k \leq K} S^k$ and according to the induction hypothesis we have $\pi'[0] \in \bigcup_{k \leq K} R^k$. Therefore, for all may successors $s'(s' = \pi'[0])$ such that $s \stackrel{may}{\longrightarrow} s'$ we have $s' \in \bigcup_{k \leq K} R^k$. This means $s \in R^{K+1}$. Therefore, we have $S^{K+1} \subseteq R^{K+1}$.

From above, we have $[(M, s) \models^3 \mathbf{AF}\phi'] = true$ iff $\varrho(R_{\mathbf{AF}\phi'})(s) = true$.

According to the assumptions and the induction hypothesis and 3-valued semantics of ALFP, we have $\{s | \varrho(R_{\mathbf{AF}\phi'})(s) \geq^3 \perp\} = \bigcup_K R^K_{tt|\perp}$, where $R^0_{tt|\perp} = \{s | \varrho(R_{\phi'})(s) \geq^3 \perp \vee \forall s' : \varrho(T)(s, s') \leq^3 \perp\} = \{s | [(M, s) \models^3 \phi'] \geq^3 \perp$ or there are no outgoing must transitions from $s\}$ and $R^{K+1}_{tt|\perp} = \{s | \forall s' : \varrho(T)(s, s') \leq^3 \perp \vee s' \in \bigcup_{k \leq K} R^k_{tt|\perp}\} \cup R^0_{tt|\perp} (K \geq 0)$.

The rest of the proof goes in two steps. We first prove that $\{s | [(M, s) \models^3 \mathbf{AF}\phi'] \geq^3 \perp\} = \bigcup_K S^K_{tt|\perp}$, where $S^K_{tt|\perp} = \{s|$ for all must path π from s: either $\exists k : 0 \leq k \leq K$ such that $[(M, \pi[k]) \models^3 \phi'] \geq^3 \perp$ or there are no outgoing must transitions from $\pi[k]\}$. Then we will prove by induction on K that $R^K_{tt|\perp} = S^K_{tt|\perp}$.

Now let's proof the first step, that is $\{s | [(M, s) \models^3 \mathbf{AF}\phi'] \geq^3 \perp\} = \bigcup_K S^K_{tt|\perp}$. Let's consider the set $T = \{s | [(M, s) \models^3 \mathbf{AF}\phi'] \geq^3 \perp\} \setminus \bigcup_K S^K_{tt|\perp}$ and we shall prove that it is empty. We proceed by contradiction. Suppose $T \neq \varnothing$ and choose $s_0 \in T$. It is obvious that $[(M, s_0) \models^3 \mathbf{AF}\phi'] \geq^3 \perp$ but $[(M, s_0) \models^3 \phi'] = false$ and there are some outgoing transitions from s_0 since otherwise $s_0 \in S^0_{tt|\perp}$(contradicting $s_0 \notin \bigcup_K S^K_{tt|\perp}$).

The transition system we consider here is finitely branching, and now we claim that for all *must successors* s_1 of $s_0(s_0 \xrightarrow{must} s_1)$, we have $[(M, s_1) \models^3 \mathbf{AF}\phi'] \geq^3 \perp$. Suppose for one must successor s_1 of s_0 we have $[(M, s_1) \models^3 \mathbf{AF}\phi'] = false$. Then there exists an infinite must path starting from s_1 ($s_1 \xrightarrow{must} s_2 \xrightarrow{must} ...$) such that for all states along the path, we have $[(M, s_i) \models^3 \phi'] = false(i \geq 1)$. Combining $s_0 \xrightarrow{must} s_1$ with the infinite may path $s_1 \xrightarrow{must} s_2 \xrightarrow{must} ...$, we get a new infinite may path $s_0 \xrightarrow{must} s_1 \xrightarrow{must} s_2 \xrightarrow{must} ...$ such that for all states along the new path, we have $[(M, s_i) \models^3 \phi'] = false(i \geq 0)$. This means $[(M, s_0) \models^3 \mathbf{AF}\phi'] = false$ and contradicts the fact that $s_0 \in T$.

On the other hand, it can't be the case that for all must successors s_1 of s_0, we have $[(M, s_1) \models^3 \phi'] \geq^3 \perp$ or there are no outgoing must transitions from s_1 since otherwise $s_0 \in S^1_{tt|\perp}$ (contradicting $s_0 \notin \bigcup_K S^K_{tt|\perp}$).

We now choose one must successor s_1 of s_0 such that $[(M, s_1) \models^3 \mathbf{AF}\phi'] \geq^3 \perp$ but $[(M, s_1) \models^3 \phi'] = false$ and there are some outgoing must transitions from s_1. Similarly, we can also show that for all must successors s_2 of s_1, we have $[(M, s_2) \models^3 \mathbf{AF}\phi'] \geq^3 \perp$. It can't be the case that for all must successors s_2 of s_1, we have $[(M, s_2) \models^3 \phi'] \geq^3 \perp$ or there are no outgoing must transitions from s_2 since otherwise $s_0 \in S^2_{tt|\perp}$ (contradicting $s_0 \notin \bigcup_K S^K_{tt|\perp}$). We can choose one must successor s_2 of s_1 such that $[(M, s_2) \models^3 \mathbf{AF}\phi'] \geq^3 \perp$ but $[(M, s_2) \models^3 \phi'] = false$ and and there are some outgoing must transitions from s_2. This process can continue arbitrarily often and produce an infinite must path starting from s_0 ($s_0 \xrightarrow{must} s_1 \xrightarrow{must} s_2 \xrightarrow{must} ...$) such that for all the states along the path, we have $[(M, s_i) \models^3 \phi'] = false(i \geq 0)$. This contradicts the assumption $[(M, s_0) \models^3 \mathbf{AF}\phi'] \geq^3 \perp$. Hence $T = \varnothing$.

For the second step, we prove $R^K_{tt|\perp} = S^K_{tt|\perp}$ by induction on K.

When $K = 0$, obviously $R^0_{tt|\perp} = S^0_{tt|\perp}$.

Let's consider $K+1$. $R^{K+1}_{tt|\perp} = \{s|\forall s' : \varrho(T)(s,s') \leq^3 \perp \vee s' \in \bigcup_{k\leq K} R^k_{tt|\perp}\} \cup R^0_{tt|\perp}(K \geq 0)$. According to the induction hypothesis, $R^{K+1} = \{s|\forall s' : \varrho(T)(s,s') \leq^3 \perp \vee s' \in \bigcup_{k\leq K} S^k_{tt|\perp}\} \cup S^0_{tt|\perp}(K \geq 0)$. It's obvious that $S^k_{tt|\perp} \subseteq S^{k+1}_{tt|\perp}(0 \leq k)$. Therefore, $S^K_{tt|\perp} = \bigcup_{k\leq K} S^k_{tt|\perp}(K \geq 0)$. Then we have $R^{K+1} = \{s|\forall s' : \varrho(T)(s,s') \leq^3 \perp \vee s' \in S^K_{tt|\perp}\} \cup S^0_{tt|\perp} = \{s|\forall s' :$ either $\varrho(T)(s,s') \leq^3 \perp$ or for all must path π from s': $\exists k : 0 \leq k \leq K$ such that $[(M,\pi[k]) \models^3 \phi'] \geq^3 \perp$ or there are no outgoing must transitions from $\pi[k]\} \cup \{s|[(M,s) \models^3 \phi'] \geq^3 \perp$ or there are no outgoing must transitions from $s\}$. According to the assumptions, we know that $\varrho(T)(s,s') = true$ iff $s \xrightarrow{must} s'$. Assume that $s \in R^{K+1}$, either we know that for all must successors s' of s ($s \xrightarrow{must} s'$) and for all must path π from s': $\exists k : 0 \leq k \leq K$ such that $[(M,\pi[k]) \models^3 \phi'] \geq^3 \perp$ or there are no outgoing transitions from $\pi[k]$, or $[(M,s) \models^3 \phi'] \geq^3 \perp$ or there are no outgoing transitions from s, or both. We have two cases. The first case is when $[(M,s) \models^3 \phi'] \geq^3 \perp$ or there are no outgoing transitions from s. Obviously we have $s \in S^{K+1}_{tt|\perp}$. The second case is when $[(M,s) \models^3 \phi'] = false$ and there are some outgoing transitions from s. In this case, for all must successors s' of s and for all must path π from s': $\exists k : 0 \leq k \leq K$ such that $[(M,\pi[k]) \models^3 \phi'] \geq^3 \perp$ or there are no outgoing transitions from $\pi[k]$. We can extend the transition $s \xrightarrow{must} s'$ to a path π' by appending the path π starting from s' to $s \xrightarrow{must} s'$ such that $\pi'[0] = s$ and $\pi'[k+1] = \pi[k](0 \leq k \leq K)$. Therefore, we know that for all must path π' from s: $\exists k : 0 \leq k \leq K+1$ such that $[(M,\pi'[k]) \models^3 \phi'] \geq^3 \perp$ or there are no outgoing transitions from $\pi[k]$. Therefore, $s \in S^{K+1}_{tt|\perp}$ as well. This proves $R^{K+1}_{tt|\perp} \subseteq S^{K+1}_{tt|\perp}$.

For the other direction, assume that $s \in S^{K+1}_{tt|\perp}$. Then for all must path π from s there exists a number $k(0 \leq k \leq K+1)$ such that $[(M,\pi[k]) \models^3 \phi'] \geq^3 \perp$ or there are no outgoing transitions from $\pi[k]$. We have two cases. The first case is when $[(M,s) \models^3 \phi'] \geq^3 \perp$ or there are no outgoing transitions from s. Obviously we have $s \in R^0_{tt|\perp}$ and therefore $s \in R^{K+1}_{tt|\perp}$. The second case is when $[(M,s) \models^3 \phi'] = false$ and there are some outgoing transitions from s. Consider the suffix π' of the path π such that $\pi'[k] = \pi[k+1](0 \leq k \leq K)$. We know that there exists a number $k(0 \leq k \leq K)$ such that $[(M,\pi'[k]) \models^3 \phi'] \geq^3 \perp$ or there are no outgoing transitions from $\pi'[k]$. This means $\pi'[0] \in S^K_{tt|\perp}$. Therefore, $\pi'[0] \in \bigcup_{k\leq K} S^k_{tt|\perp}$ and according to the induction hypothesis we have $\pi'[0] \in \bigcup_{k\leq K} R^k_{tt|\perp}$. Therefore, for all must successors $s'(s' = \pi'[0])$ such that $s \xrightarrow{must} s'$ we have $s' \in \bigcup_{k\leq K} R^k_{tt|\perp}$. According to the assumptions, we know that $\varrho(T)(s,s') = true$ iff $s \xrightarrow{must} s'$. Therefore, for all must successors $s'(s' = \pi'[0])$ such that $\varrho(T)(s,s') = true$ we have $s' \in \bigcup_{k\leq K} R^k_{tt|\perp}$. This means $s \in R^{K+1}_{tt|\perp}$. Therefore, we have $S^{K+1}_{tt|\perp} \subseteq R^{K+1}_{tt|\perp}$.

From above, we have $[(M, s) \models^3 \mathbf{AF}\phi'] \geq^3 \perp$ iff $\varrho(R_{\mathbf{AF}\phi'})(s) \geq^3 \perp$. $\square$

Appendix C

Appendix for Chapter 5

Lemma 5.4 Let ϕ be an alternation-free μ-calculus formula in Negation-free PNF and assume that we translate ϕ to its Alternation-free Normal Form ϕ' using our translation method. Then, each subformula of the form $\neg\mu Q.\varphi$ in the formula ϕ' is indeed closed and no negations are applied to variables in ϕ'.

Proof. We first point out the following fact that all negative occurrence of μ operators in ϕ' are generated only in the following three cases.

1. When using the duality to eliminate the ν operator for all top ν-subformulas $\nu Q'.\varphi$ of ϕ, the main connective (negation) of the resulting equivalent formula $\neg\mu Q'.\neg\varphi[\neg Q'/Q']$ will remain there and can not be pushed deeper any further.
2. When using the duality to eliminate the ν operator for all top-level ν-subformulas $\nu Q'.\varphi$ of any μ-subformula $\mu Q''.\varphi'$ of ϕ, the main connective (negation) of the resulting equivalent formula $\neg\mu Q'.\neg\varphi[\neg Q'/Q']$ will remain there and can not be pushed deeper any further.
3. When eliminating a ν operator for any ν-subformulas $\nu Q'.\varphi$ of ϕ by duality, a negation will be pushed to all the top μ-subformulas $\mu Q''.\varphi'$ of $\nu Q'.\varphi$ by De Morgan's law and other dualities. The negation will remain in front of $\mu Q''.\varphi'$ and can not be pushed deeper any further.

It is straightforward to see from the above mentioned fact why any subformula of the form $\neg\mu Q.\varphi$ in formula ϕ' are indeed closed. In the first case, if the resulting equivalent formula $\neg\mu Q'.\neg\varphi[\neg Q'/Q']$ is not closed, the corresponding top ν-subformulas $\nu Q'.\varphi$ of ϕ cannot be closed either. This contradicts our assumption that ϕ is closed. In the last two cases, the resulting equivalent formulas should also be closed, otherwise it contradicts Lemma 5.3. Notice that a top μ-subformula is also a top-level μ-subformula.

Let's go back to the third case in the fact mentioned above. Assume that $\mu Q''.\varphi'$ is a top μ-subformula of the ν-subformula $\nu Q'.\varphi$ of ϕ. After using duality $\nu Q'.\varphi \equiv \neg\mu Q'.\neg\varphi[\neg Q'/Q']$, negations are applied to all occurrence of Q' in φ now. If $\mu Q''.\varphi'$ is not closed and contains Q', the negations applied to some of the occurrences of Q' in φ' can not be eliminated. Since we have proved that in ϕ' any subformula of the form $\neg\mu Q.\varphi$ is closed, this avoids the only possibility that a negation can be applied to variables. $\square$

Lemma 5.6 Given a μ-calculus formula ϕ' in Negation-free PNF which is translated from a closed formula ϕ in Alternation-free Normal Form. Assume that $\phi'_1,\ldots,\phi'_n$ are the maximal formulas of the set of closed proper fixpoint subformulas of ϕ', the alternation depth of ϕ'', which is obtained from ϕ' by substituting new atomic propositions $p_1,\ldots,p_n$ for $\phi'_1,\ldots,\phi'_n$, is strictly less than 2.

Proof. The most interesting cases are $\phi = \mu Q.\varphi$ and $\phi = \neg\mu Q.\varphi$.

1. $\phi = \mu Q.\varphi$: It is obvious that there are only μ-subformulas in ϕ''. According to definition 5.1, $ad(\phi'') < 2$.
2. $\phi = \neg\mu Q.\varphi$: It is not difficult to see that after the translation there are only ν-subformlas in ϕ''. According to definition 5.1, $ad(\phi'') < 2$.
3. If ϕ does not belong to the above two cases, then ϕ'' actually doesn't contain any fixpoint subformulas. This trivially means $ad(\phi'') < 2$.

$\square$

Lemma 5.7 Every μ-calculus formula ϕ' in Negation-free PNF translated from a closed formula ϕ in Alternation-free Normal Form is alternation-free.

Proof. We give an informal analysis of the process of calculating the alternation depth of ϕ' below. Assume that ϕ'_1,...,ϕ'_n are the maximal formulas of the set of closed proper fixpoint subformulas of ϕ'. According to Definition 5.1, we have the following:

$$ad(\phi') = max(ad(\phi''), ad(\phi'_1), ..., ad(\phi'_n))$$

where ϕ'' is obtained from ϕ' by substituting new atomic propositions p_1,...,p_n for ϕ'_1,...,ϕ'_n.

From Lemma 5.6, we know that $ad(\phi'') < 2$. Whether the alternation depth of ϕ' is less than 2 or not now depends on $ad(\phi'_i)$ where $0 \leq i \leq n$. Actually, for each subformula ϕ'_i, there is a corresponding closed μ-subformula ϕ_i in ϕ such that ϕ'_i is either translated directly from ϕ_i or from $\neg\phi_i$. Notice that both ϕ_i and $\neg\phi_i$ are closed and are in Alternation-free Normal Form. Therefore, the problem of calculating the alternation depth of ϕ' has been "reduced" to a simpler problem of calculating the alternation depth of each ϕ'_i . This problem-reduction process can continue again and again. Finally the problem will be reduced to calculating the alternation depth of a closed fixpoint subformula φ'_i of ϕ', where φ'_i is translated from φ_i or $\neg\varphi_i$ and φ_i is a closed μ-subformula of ϕ without any closed proper fixpoint subformulas. It's obvious that φ'_i has no closed proper fixpoint subformulas. We can know from Lemma 5.6 that $ad(\varphi'_i) < 2$. Therefore we know that $ad(\phi') < 2$ which means ϕ' is alternation-free. □

Theorem 5.11 Given a μ-calculus formula ϕ in Alternation-free Normal Form with $Q_1, ..., Q_n$ being all the free variables in it and assume that $\phi \longmapsto \langle cl_\phi,\ pre_\phi\rangle$. We know that $s' \in [\![\phi]\!]_{e[Q_1\mapsto S_1,...,Q_n\mapsto S_n]}$ iff $(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_ϕ for the least solution ϱ of cl_ϕ subject to $\varrho(R_{Q_1}) \supseteq S_1, ..., \varrho(R_{Q_n}) \supseteq S_n, \varrho \supseteq \varrho_0$ ($\varrho = \sqcap\{\varrho' \mid (\varrho', \sigma)$ $\underline{\texttt{sat}}$ $cl_\phi \wedge \varrho'(R_{Q_1}) \supseteq S_1, ..., \wedge\ \varrho'(R_{Q_n}) \supseteq S_n, \varrho \supseteq \varrho_0\}$), where ϱ_0 defines P_p and T_a.

Proof. We proceed by structural induction on ϕ.

Case $\phi = p$: According to the semantics of μ-calculus, we know that $s' \in [\![p]\!]$ iff $p \in L(s')$. We know from the least solution ϱ of **true** subject to $\varrho \supseteq \varrho_0$ that $\varrho = \varrho_0$. Therefore, we know that $(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $P_p(s)$ iff $s' \in \varrho(P_p)$ iff $s' \in \varrho_0(P_p)$ iff $p \in L(s')$. This means $s' \in [\![p]\!]$ iff $(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $P_p(s)$ for the least solution ϱ of **true** subject to $\varrho \supseteq \varrho_0$.

Case $\phi = Q$: Let $e' = e[Q \mapsto S]$. According to the semantics of μ-calculus, we know that $s' \in [\![Q]\!]_{e'}$ iff $s' \in e'(Q)$ iff $s' \in S$. From the least solution ϱ of **true** subject to $\varrho(R_Q) \supseteq S, \varrho \supseteq \varrho_0$, we know that that $\varrho(R_Q) = S$. Therefore, we know that $(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $R_Q(s)$ iff $s' \in \varrho(R_Q)$ iff $s' \in S$. This means $s' \in [\![Q]\!]_{e'}$ iff $(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $R_Q(s)$ for the least solution ϱ of **true** subject to $\varrho(R_Q) \supseteq S, \varrho \supseteq \varrho_0$.

Case $\phi = \phi_1 \vee \phi_2$: Assume that $Q_1, ..., Q_n$ are all the free variables in $\phi_1 \vee \phi_2$. Let $e' = e[Q_1 \mapsto S_1, ..., Q_n \mapsto S_n]$. Let's consider the least model ϱ for $cl_{\phi_1} \wedge cl_{\phi_2}$ subject to $\varrho(R_{Q_1}) \supseteq S_1, ..., \varrho(R_{Q_n}) \supseteq S_n, \varrho \supseteq \varrho_0$. It is possible that ϕ_1 and ϕ_2 have a same subformula. In this case, we map the same formula in ϕ_1 and ϕ_2 in the same way according to Table 5.1. Assume that R_Q is defined in cl_{ϕ_1} (or cl_{ϕ_1}), where $\mu Q.\varphi$ is a subformula of ϕ_1 (or ϕ_2), we know that the relation $\varrho(R_Q)$ coincides with the relation $\varrho'(R_Q)$ in the least model ϱ' for cl_{ϕ_1} (or cl_{ϕ_2}) subject to $\varrho'(R_{Q_1}) \supseteq S_1, ..., \varrho'(R_{Q_n}) \supseteq S_n, \varrho' \supseteq \varrho_0$. Therefore, we know that for a given s', $(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_{ϕ_1} iff $(\varrho', \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_{ϕ_1} and that $(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_{ϕ_2} iff $(\varrho', \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_{ϕ_2}.

According to the semantics of μ-calculus, $s' \in [\![\phi_1 \vee \phi_2]\!]_{e'}$ iff $s' \in [\![\phi_1]\!]_{e'}$ or $s' \in [\![\phi_2]\!]_{e'}$ holds. According to the induction hypothesis, we know that $s' \in [\![\phi_1]\!]_{e'}$ iff $(\varrho', \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_{ϕ_1} and that $s' \in [\![\phi_2]\!]_{e'}$ iff $(\varrho', \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_{ϕ_2}. According to the semantics of ALFP, we know that $(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_{\phi_1} \vee pre_{\phi_2}$ iff $(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_{ϕ_1} or $(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_{ϕ_2} holds. Therefore, $s' \in [\![\phi_1 \vee \phi_2]\!]_{e'}$ iff $(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_{\phi_1} \vee pre_{\phi_2}$ in the least model ϱ for $cl_{\phi_1} \wedge cl_{\phi_2}$ subject to $\varrho(R_{Q_1}) \supseteq S_1, ..., \varrho(R_{Q_n}) \supseteq S_n, \varrho \supseteq \varrho_0$.

Case $\phi = \phi_1 \wedge \phi_2$: This case is similar to $\phi = \phi_1 \vee \phi_2$.

Case $\phi = \langle a \rangle \varphi$: Assume that $Q_1, ..., Q_n$ are all the free variables in $\langle a \rangle \varphi$. Let $e' = e[Q_1 \mapsto S_1, ..., Q_n \mapsto S_n]$. Let's consider the least model ϱ for $cl_{\langle a \rangle \varphi}$ subject to $\varrho(R_{Q_1}) \supseteq S_1, ..., \varrho(R_{Q_n}) \supseteq S_n, \varrho \supseteq \varrho_0$. Assume that R_Q is defined in cl_φ, where $\mu Q.\varphi'$ is a subformula of φ, we know that the relation $\varrho(R_Q)$ coincides with the relation $\varrho'(R_Q)$ in the least model ϱ' for cl_φ subject to $\varrho'(R_{Q_1}) \supseteq S_1, ..., \varrho'(R_{Q_n}) \supseteq S_n, \varrho' \supseteq \varrho_0$. Therefore, we know that for a given s', $(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_φ iff $(\varrho', \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_φ.

According to the semantics of μ-calculus, $s'' \in [\![\langle a \rangle \varphi]\!]_{e'}$ iff $\exists s' : (s'', s') \in a \wedge s' \in [\![\varphi]\!]_{e'}$ holds. Notice that $(s'', s') \in \varrho(T_a)$ iff $(s'', s') \in a$. According to the induction hypothesis, we know that $s' \in [\![\varphi]\!]_{e'}$ iff $(\varrho', \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_φ. According to the semantics of ALFP, $(\varrho, \sigma[s \mapsto s''])$ $\underline{\texttt{sat}}$ $\exists s' : T_a(s,s') \wedge pre_\phi[s'/s]$ iff $(\varrho, \sigma[s \mapsto s'', s' \mapsto t])$ $\underline{\texttt{sat}}$ $T_a(s,s') \wedge pre_\phi[s'/s]$ holds for some $t \in S$. Therefore, it's easy to see that $s'' \in [\![\langle a \rangle \varphi]\!]_{e'}$ iff $(\varrho, \sigma[s \mapsto s''])$ $\underline{\texttt{sat}}$ $\exists s' : T_a(s,s') \wedge pre_\phi[s'/s]$ in the least model ϱ for $cl_{\langle a \rangle \varphi}$ subject to $\varrho(R_{Q_1}) \supseteq S_1, ..., \varrho(R_{Q_n}) \supseteq S_n, \varrho \supseteq \varrho_0$.

Case $\phi = [a]\varphi$: Assume that $Q_1, ..., Q_n$ are all the free variables in $[a]\varphi$. Let $e' = e[Q_1 \mapsto S_1, ..., Q_n \mapsto S_n]$. Let's consider the least model ϱ for $cl_{[a]\varphi}$ subject to $\varrho(R_{Q_1}) \supseteq S_1, ..., \varrho(R_{Q_n}) \supseteq S_n, \varrho \supseteq \varrho_0$. Assume that R_Q is defined in cl_φ, where $\mu Q.\varphi'$ is a subformula of φ, we know that the relation $\varrho(R_Q)$ coincides with the relation $\varrho'(R_Q)$ in the least model ϱ' for cl_φ subject to $\varrho'(R_{Q_1}) \supseteq S_1, ..., \varrho'(R_{Q_n}) \supseteq S_n, \varrho' \supseteq \varrho_0$. Therefore, we know that for a given s', $(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_φ iff $(\varrho', \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_φ.

According to the semantics of μ-calculus, $s'' \in [\![[a]\varphi]\!]_{e'}$ iff $\forall s' : (s'', s') \in a$ implies $s' \in [\![\varphi]\!]_{e'}$ iff $\forall s' : (s'', s') \notin a \vee s' \in [\![\varphi]\!]_{e'}$. Notice that $(s'', s') \in \varrho(T_a)$ iff $(s'', s') \in a$. According to the induction hypothesis, we know that $s' \in [\![\varphi]\!]_{e'}$ iff $(\varrho', \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_φ. According to the semantics of ALFP, $(\varrho, \sigma[s \mapsto s''])$ $\underline{\texttt{sat}}$ $\forall s' : \neg T_a(s,s') \vee pre_\phi[s'/s]$ iff $(\varrho, \sigma[s \mapsto s'', s' \mapsto t])$ $\underline{\texttt{sat}}$ $\neg T_a(s,s') \vee pre_\phi[s'/s]$ holds for all $t \in S$. Therefore, it's easy to see that $s'' \in [\![[a]\varphi]\!]_{e'}$ iff $(\varrho, \sigma[s \mapsto s''])$ $\underline{\texttt{sat}}$ $\forall s' : \neg T_a(s,s') \vee pre_\phi[s'/s]$ in the least model ϱ for $cl_{[a]\varphi}$ subject to $\varrho(R_{Q_1}) \supseteq S_1, ..., \varrho(R_{Q_n}) \supseteq S_n, \varrho \supseteq \varrho_0$.

Case $\phi = \neg\mu Q.\varphi$: Notice that in the syntax of Alternation-free Normal Form, the formula $\neg\mu Q.\varphi$ is closed. Let's consider the least model ϱ for $cl_{\neg\mu Q.\varphi}$ subject to $\varrho \supseteq \varrho_0$. We know that the relation $\varrho(R_Q)$ coincides with the relation $\varrho'(R_Q)$ in the least model ϱ' for $cl_{\mu Q.\varphi}$ subject to $\varrho' \supseteq \varrho_0$. Therefore, we know that for a given s', $(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $R_Q(s)$ iff $(\varrho', \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $R_Q(s)$. That is $\varrho(R_Q) = \varrho'(R_Q)$.

According to the semantics of μ-calculus, $s' \in [\![\neg\mu Q.\varphi]\!]$ iff $s' \notin [\![\mu Q.\varphi]\!]$. According to the induction hypothesis, we know that $s' \in [\![\mu Q.\varphi]\!]$ iff $(\varrho', \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $R_Q(s)$ iff $s' \in \varrho'(R_Q)$. According to the semantics of ALFP, $(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $\neg R_Q(s)$ iff $s' \notin \varrho(R_Q)$. Therefore, $s' \in [\![\neg\mu Q.\varphi]\!]$ iff $(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $\neg R_Q(s)$ in the least model ϱ for $cl_{\neg\mu Q.\varphi}$ subject to $\varrho \supseteq \varrho_0$.

Case $\phi = \mu Q.\varphi$: Assume that $Q_1, ..., Q_n$ are all the free variables in $\mu Q.\varphi$. Let $e' = e[Q_1 \mapsto S_1, ..., Q_n \mapsto S_n]$. The intuition of our proof is that we want to show $\varrho_1 = \varrho_2$ where ϱ_1 is the least model for $cl_{\mu Q.\varphi}$ subject to $\varrho_1(R_{Q_1}) \supseteq S_1, ..., \varrho_1(R_{Q_n}) \supseteq S_n, \varrho \supseteq \varrho_0$ and ϱ_2 is a least model constructed in a way that mimics the μ-calculus semantics of $\mu Q.\varphi$. This will show that the analysis result of our approach matches the μ-calculus semantics in the case of $\phi = \mu Q.\varphi$, which means $\varrho_1(R_Q) = [\![\mu Q.\varphi]\!]$ holds. Therefore, $s' \in [\![\mu Q.\varphi]\!]_{e'}$ iff $(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $R_Q(s)$ in the least model ϱ for $cl_{\mu Q.\varphi}$ subject to $\varrho(R_{Q_1}) \supseteq S_1, ..., \varrho(R_{Q_n}) \supseteq S_n, \varrho \supseteq \varrho_0$. The models ϱ_1 and ϱ_2 are defined as follows.

The model ϱ_1 is defined by $\varrho_1 = \sqcap\Psi_1$ where $\Psi_1 = \{\varrho \mid \varrho \models cl_\varphi \wedge \{s'|(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_\varphi\} \subseteq \varrho(R_Q) \wedge \varrho(R_{Q_1}) \supseteq S_1, ..., \wedge\ \varrho(R_{Q_n}) \supseteq S_n \wedge \varrho \supseteq \varrho_0\}$. The model ϱ_2 is defined by $\varrho_2 = \sqcap\Psi_2$ where $\Psi_2 = \{\varrho \mid \varrho = \sqcap\{\varrho' \mid \varrho' \models cl_\varphi \wedge\ S' \subseteq \varrho'(R_Q)\ \wedge \varrho'(R_{Q_1}) \supseteq S_1, ..., \wedge\ \varrho'(R_{Q_n}) \supseteq S_n \wedge \varrho' \supseteq \varrho_0\}\ \wedge\ \{s'|(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_\varphi\} \subseteq S'\ \wedge S' \subseteq \mathcal{U}\}$. It's not difficult to prove that Ψ_1 is a Moore Family and $\varrho_2 \in \Psi_2$. Therefore, ϱ_1 is an element of Ψ_1 and ϱ_2 is an element of Ψ_2.

Our main idea of proving that $\varrho_1 = \varrho_2$ holds is as follows. We first show that $\varrho_2 \in \Psi_1$. If this is true, then we know that $\varrho_1 \sqsubseteq \varrho_2$ since ϱ_1 is a lower bound of Ψ_1. Second, we show that $\varrho_1 \in \Psi_2$. If this holds, similarly we know that $\varrho_2 \sqsubseteq \varrho_1$ since ϱ_2 is a lower bound of Ψ_2. Then, since $\sqsubseteq$ is anti-symmetric, we have that $\varrho_1 = \varrho_2$.

First, we try to show that $\varrho_2 \in \Psi_1$ holds. It is obvious that $\varrho_2 \models cl_\varphi \wedge \{s'|(\varrho_2, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_\varphi\} \subseteq \varrho_2(R_Q) \wedge \varrho_2(R_{Q_1}) \supseteq S_1, ..., \wedge\ \varrho_2(R_{Q_n}) \supseteq S_n \wedge \varrho_2 \supseteq \varrho_0$ holds. Therefore, we have $\varrho_2 \in \Psi_1$.

Now, we try to show that $\varrho_1 \in \Psi_2$ also holds. Restricting the S' in the definition of Ψ_2 to the particular value $\{s'|(\varrho_1, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_\varphi\}$, we get the subset Ψ_2' of Ψ_2 where $\Psi_2' = \{\varrho \mid \varrho = \sqcap\{\varrho' \mid \varrho' \models cl_\varphi \wedge\ \{s'|(\varrho_1, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_\varphi\} \subseteq \varrho'(R_Q)\ \wedge \varrho'(R_{Q_1}) \supseteq S_1, ..., \wedge\ \varrho'(R_{Q_n}) \supseteq S_n \wedge \varrho' \supseteq \varrho_0\}\ \wedge \{s'|(\varrho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_\varphi\} \subseteq \{s'|(\varrho_1, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_\varphi\}\ \wedge \{s'|(\varrho_1, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_\varphi\} \subseteq \mathcal{U}\}$. Obviously Ψ_2' is a Moore Family. If $\varrho_1 \in \Psi_2'$ holds, $\varrho_1 \in \Psi_2$ also holds. To show that $\varrho_1 \in \Psi_2'$ holds, we need to prove that $\varrho_1 = \sqcap\{\varrho' \mid \varrho' \models cl_\varphi \wedge\ \{s'|(\varrho_1, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_\varphi\} \subseteq \varrho'(R_Q)\ \wedge \varrho'(R_{Q_1}) \supseteq S_1, ..., \wedge\ \varrho'(R_{Q_n}) \supseteq S_n \wedge \varrho' \supseteq \varrho_0\} \wedge \{s'|(\varrho_1, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_\varphi\} \subseteq \{s'|(\varrho_1, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_\varphi\}\ \wedge \{s'|(\varrho_1, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_\varphi\} \subseteq \mathcal{U}$ holds. It is obvious that $\{s'|(\varrho_1, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_\varphi\} \subseteq \{s'|(\varrho_1, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_\varphi\}\ \wedge \{s'|(\varrho_1, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_\varphi\} \subseteq \mathcal{U}$ holds. It remains to prove that $\varrho_1 = \sqcap\{\varrho' \mid \varrho' \models cl_\varphi \wedge \{s'|(\varrho_1, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_\varphi\} \subseteq \varrho'(R_Q) \wedge \varrho'(R_{Q_1}) \supseteq S_1, ..., \wedge\ \varrho'(R_{Q_n}) \supseteq S_n \wedge \varrho' \supseteq \varrho_0\}$ holds. We prove this equation as follows.

Let $\varrho' = \sqcap\Psi'$ where $\Psi' = \{\varrho' \mid \varrho' \models\ cl_\varphi \wedge\ \{s'|(\varrho_1, \sigma[s \mapsto s'])\ \underline{\texttt{sat}}\ pre_\varphi\} \subseteq \varrho'(R_Q)\ \wedge \varrho'(R_{Q_1}) \supseteq S_1, ..., \wedge\ \varrho'(R_{Q_n}) \supseteq S_n \wedge \varrho' \supseteq \varrho_0\}$ and now the equation we want to prove becomes $\varrho_1 = \varrho'$.

Since $\varrho_1 \models\ cl_\varphi \wedge\ \{s'|(\varrho_1, \sigma[s \mapsto s'])\ \underline{\texttt{sat}}\ pre_\varphi\} \subseteq \varrho_1(R_Q)\ \wedge \varrho_1(R_{Q_1}) \supseteq S_1, ..., \wedge\ \varrho_1(R_{Q_n}) \supseteq S_n \wedge \varrho_1 \supseteq \varrho_0$ holds, we know that $\varrho_1 \in \Psi'$. Since ϱ' is a lower bound of Ψ', $\varrho' \sqsubseteq \varrho_1$ holds. We know that ϱ_1 is the least element in Ψ_1 and ϱ' is the least element in Ψ'. We can know from the definition of Ψ_1 and Ψ' that $\varrho_1(R_{Q'}) = \varrho'(R_{Q'})$ for all $R_{Q'}$ such that $\texttt{rank}_{R_{Q'}} < \texttt{rank}_{R_Q}$. Therefore, from $\varrho' \sqsubseteq \varrho_1$ and the definition of lexicographic ordering, we know that $\varrho'(R_{Q''}) \subseteq \varrho_1(R_{Q''})$ for all $R_{Q''}$ such that $\texttt{rank}_{R_{Q''}} = \texttt{rank}_{R_Q}$. Now it is not difficult to prove that $\{s'|(\varrho', \sigma[s \mapsto s'])\ \underline{\texttt{sat}}\ pre_\varphi\} \subseteq \{s'|(\varrho_1, \sigma[s \mapsto s'])\ \underline{\texttt{sat}}\ pre_\varphi\}$ holds, (i.e. Lemma D.1 helps to proof this statement). This means $\{s'|(\varrho', \sigma[s \mapsto s'])\ \underline{\texttt{sat}}\ pre_\varphi\} \subseteq \varrho'(R_Q)$ holds. Therefore, we know that $\varrho' \models cl_\varphi \wedge\ \{s'|(\varrho_1, \sigma[s \mapsto s'])\ \underline{\texttt{sat}}\ pre_\varphi\} \subseteq \varrho'(R_Q)\ \wedge \varrho'(R_{Q_1}) \supseteq S_1, ..., \wedge\ \varrho'(R_{Q_n}) \supseteq S_n \wedge \varrho' \supseteq \varrho_0$ holds. This means $\varrho' \in \Psi_1$. Therefore, $\varrho_1 \sqsubseteq \varrho'$. Since $\sqsubseteq$ is anti-symmetric, we know that $\varrho_1 = \varrho'$.

According to induction hypothesis, $[\![\varphi]\!]_{e'[Q\mapsto S']}$ equals the set $\{s'|(\varrho', \sigma[s \mapsto s'])\ \underline{\texttt{sat}}\ pre_\varphi\}$ in the least model ϱ' for cl_φ subject to $\varrho'(R_{Q_1}) \supseteq S_1, ..., \varrho'(R_{Q_n}) \supseteq S_n, \varrho'(R_Q) \supseteq S', \varrho' \supseteq \varrho_0$. Therefore, we know that $\varrho_2(R_Q) = \cap\{S' \subseteq \mathcal{U}| [\![\varphi]\!]_{e'[Q\mapsto S']} \subseteq S'\}$. This is exactly the least fixed point of the monotone function $\tau(\omega) = [\![\varphi]\!]_{e'[Q\mapsto\omega]}$, which means $\varrho_2(R_Q) = [\![\mu Q.\varphi]\!]_{e'}$. Since we have proved that $\varrho_1 = \varrho_2$, we know that $[\![\mu Q.\varphi]\!]_{e'} = \varrho_1(R_Q)$. Therefore, $s' \in [\![\mu Q.\varphi]\!]_{e'}$ iff $(\varrho, \sigma[s \mapsto s'])\ \underline{\texttt{sat}}\ R_Q(s)$ in the least model ϱ for $cl_{\mu Q.\varphi}$ subject to $\varrho(R_{Q_1}) \supseteq S_1, ..., \varrho(R_{Q_n}) \supseteq S_n, \varrho \supseteq \varrho_0$. $\square$

Appendix D

Appendix for Chapter 6

Lemma D.1 *Given a negation-free precondition pre and two interpretations ρ_1 and ρ_2 such that $\rho_1(R) \subseteq \rho_2(R)$ where R occurs in pre, we then have that (ρ_1, σ) $\underline{\texttt{sat}}$ pre implies (ρ_2, σ) $\underline{\texttt{sat}}$ pre.*

Proof. We prove by induction on *pre*.

Case $pre = R(v_1, ..., v_n)$: Assume that (ρ_1, σ) $\underline{\texttt{sat}}$ $R(v_1, ..., v_n)$ holds. According to the semantics for *pre*, we know that $(\sigma(v_1), ..., \sigma(v_n)) \in \rho_1(R)$ also holds. From $\rho_1 \subseteq \rho_2$, we know that $\rho_1(R) \subseteq \rho_2(R)$. Therefore, $(\sigma(v_1), ..., \sigma(v_n)) \in \rho_2(R)$. According to the semantics for *pre*, (ρ_2, σ) $\underline{\texttt{sat}}$ $R(v_1, ..., v_n)$ also holds.

Case $pre = pre_1 \wedge pre_2$: Assume that (ρ_1, σ) $\underline{\texttt{sat}}$ $pre_1 \wedge pre_2$. According to the semantics for *pre*, we know that (ρ_1, σ) $\underline{\texttt{sat}}$ pre_1 and (ρ_1, σ) $\underline{\texttt{sat}}$ pre_2 also hold. Since $pre_1 \wedge pre_2$ is negation-free, we know that both pre_1 and pre_2 are negation-free. According to the induction hypothesis, we know that (ρ_2, σ) $\underline{\texttt{sat}}$ pre_1 and (ρ_2, σ) $\underline{\texttt{sat}}$ pre_2. Therefore, we know that (ρ_2, σ) $\underline{\texttt{sat}}$ $pre_1 \wedge pre_2$ holds.

Case $pre = pre_1 \vee pre_2$: Assume that (ρ_1, σ) $\underline{\texttt{sat}}$ $pre_1 \vee pre_2$. According to the semantics for pre, we know that either (ρ_1, σ) $\underline{\texttt{sat}}$ pre_1 or (ρ_1, σ) $\underline{\texttt{sat}}$ pre_2 holds. Since $pre_1 \vee pre_2$ is negation-free, we know that both pre_1 and pre_2 are negation-free. According to the induction hypothesis, we know that either (ρ_2, σ) $\underline{\texttt{sat}}$ pre_1 or (ρ_2, σ) $\underline{\texttt{sat}}$ pre_2 holds. Therefore, we know that (ρ_2, σ) $\underline{\texttt{sat}}$ $pre_1 \vee pre_2$ holds.

Case $pre = \forall x : pre'$: Assume that (ρ_1, σ) $\underline{\texttt{sat}}$ $\forall x : pre'$ holds. According to the semantics for pre, we know that $(\rho_1, \sigma[x \mapsto a])$ $\underline{\texttt{sat}}$ pre' for all $a \in \mathcal{U}$. Since $\forall x : pre'$ is negation-free, we know that pre' is also negation-free. According to the induction hypothesis, we know that $(\rho_2, \sigma[x \mapsto a])$ $\underline{\texttt{sat}}$ pre' for all $a \in \mathcal{U}$. Therefore, (ρ_2, σ) $\underline{\texttt{sat}}$ $\forall x : pre'$ also holds.

Case $pre = \exists x : pre'$: Assume that (ρ_1, σ) $\underline{\texttt{sat}}$ $\exists x : pre'$ holds. According to the semantics for pre, we know that $(\rho_1, \sigma[x \mapsto a])$ $\underline{\texttt{sat}}$ pre' for some $a \in \mathcal{U}$. Since $\exists x : pre'$ is negation-free, we know that pre' is also negation-free. According to the induction hypothesis, we know that $(\rho_2, \sigma[x \mapsto a])$ $\underline{\texttt{sat}}$ pre' for some $a \in \mathcal{U}$. Therefore, (ρ_2, σ) $\underline{\texttt{sat}}$ $\exists x : pre'$ also holds. □

Lemma D.2 *Given a negation-free clause cl. Let ρ_1 and ρ_2 be two interpretations such that $\rho_1(R) = \rho_2(R)$ where R is defined in cl and that $\rho_2(R') \subseteq \rho_1(R')$ where R' occurs in cl but is not defined in cl. Then we have (ρ_1, σ) $\underline{\texttt{sat}}$ cl implies (ρ_2, σ) $\underline{\texttt{sat}}$ cl.*

Proof. We prove by induction on cl.

Case $cl = R(v_1, ..., v_n)$: Assume that (ρ_1, σ) $\underline{\texttt{sat}}$ $R(v_1, ..., v_n)$ holds. According to the semantics for cl, we know that $(\sigma(v_1), ..., \sigma(v_n)) \in \rho_1(R)$. Since R is defined in cl, we know that $(\sigma(v_1), ..., \sigma(v_n)) \in \rho_2(R)$. Therefore, we have that (ρ_2, σ) $\underline{\texttt{sat}}$ $R(v_1, ..., v_n)$.

Case $cl = \mathbf{true}$: This case is trivial, since (ρ_2, σ) $\underline{\texttt{sat}}$ $\mathbf{true}$ always holds.

Case $cl = pre \Rightarrow R(v_1, ..., v_n)$: Assume that (ρ_1, σ) $\underline{\texttt{sat}}$ $pre \Rightarrow R(v_1, ..., v_n)$ holds. According to the semantics for cl, we know that (ρ_1, σ) $\underline{\texttt{sat}}$ $R(v_1, ..., v_n)$ whenever (ρ_1, σ) $\underline{\texttt{sat}}$ pre. We now have the following two cases.

Assume that (ρ_1, σ) $\underline{\texttt{sat}}$ $R(v_1, ..., v_n)$. This means $(\sigma(v_1), ..., \sigma(v_n)) \in \rho_1(R)$.

Since R is defined in cl, we know that $(\sigma(v_1), ..., \sigma(v_n)) \in \rho_2(R)$. Therefore, we have that (ρ_2, σ) $\underline{\texttt{sat}}$ $R(v_1, ..., v_n)$. According to the semantics for cl, we know that (ρ_2, σ) $\underline{\texttt{sat}}$ $pre \Rightarrow R(v_1, ..., v_n)$ also holds.

Assume that (ρ_1, σ) $\underline{\texttt{sat}}$ $R(v_1, ..., v_n)$ does not hold. In this case, we know that (ρ_1, σ) $\underline{\texttt{sat}}$ pre should not hold. We will prove by contradiction that (ρ_2, σ) $\underline{\texttt{sat}}$ pre does not hold. From the definition of ρ_1 and ρ_2, we know that $\rho_2 \subseteq \rho_1$. Assume that (ρ_2, σ) $\underline{\texttt{sat}}$ pre holds. From Lemma D.1, we know that (ρ_1, σ) $\underline{\texttt{sat}}$ pre should also hold. This is a contradiction. Therefore, we know that (ρ_2, σ) $\underline{\texttt{sat}}$ pre does not hold. According to the semantics for cl, we know that (ρ_2, σ) $\underline{\texttt{sat}}$ $pre \Rightarrow R(v_1, ..., v_n)$ holds.

Case $cl = cl_1 \wedge cl_2$: Assume that (ρ_1, σ) $\underline{\texttt{sat}}$ $cl_1 \wedge cl_2$ holds. According to the semantics for cl, we know that (ρ_1, σ) $\underline{\texttt{sat}}$ cl_1 and (ρ_1, σ) $\underline{\texttt{sat}}$ cl_2. Since cl is negation-free, we know that both cl_1 and cl_2 are also negation-free. According to the induction hypothesis, we know that (ρ_2, σ) $\underline{\texttt{sat}}$ cl_1 and (ρ_2, σ) $\underline{\texttt{sat}}$ cl_2. Therefore, (ρ_2, σ) $\underline{\texttt{sat}}$ $cl_1 \wedge cl_2$ also holds.

Case $cl = \forall x : cl'$: Assume that (ρ_1, σ) $\underline{\texttt{sat}}$ $\forall x : cl'$ holds. According to the semantics for cl, we know that $(\rho_1, \sigma[x \mapsto a])$ $\underline{\texttt{sat}}$ cl' for all $a \in \mathcal{U}$. Since $\forall x : cl'$ is negation-free, we know that cl' is also negation-free. According to the induction hypothesis, we know that $(\rho_2, \sigma[x \mapsto a])$ $\underline{\texttt{sat}}$ cl' for all $a \in \mathcal{U}$. Therefore, (ρ_2, σ) $\underline{\texttt{sat}}$ $\forall x : cl'$ also holds. $\square$

Lemma D.3 *Let* $cls = cl_1, ..., cl_n$ *be weakly stratified and* $1 \leq i, j \leq n$. *Let* $\rho_1 = \varrho_0^1, ..., \varrho_n^1$ *and* $\rho_2 = \varrho_0^2, ..., \varrho_n^2$ *be two interpretations such that* 1) $\varrho_i^1 = \varrho_i^2$, 2) *if* cl_i *depends positively on* cl_j *when* $i \neq j$, *then* $\varrho_j^2 \subseteq \varrho_j^1$, *and* 3) *if* cl_i *depends negatively on* cl_j, *then* $\varrho_j^2 \supseteq \varrho_j^1$. *Then we have* (ρ_1, σ) $\underline{\texttt{sat}}$ cl_i *implies* (ρ_2, σ) $\underline{\texttt{sat}}$ cl_i.

Proof. Notice that we can transform cl_i to a negation-free clause cl_i' by substituting all negative queries of the form $\neg R$ in cl_i with a relation $R^\neg$. Let ρ be an interpretation. We interpret $R^\neg$ in ρ by defining that $\rho(R^\neg) = \neg\rho(R)$. It's obvious that (ρ, σ) $\underline{\texttt{sat}}$ cl_i iff (ρ, σ) $\underline{\texttt{sat}}$ cl_i'.

We now interpret $R^\neg$ in ρ_1 (resp. ρ_2) by defining that $\rho_1(R^\neg) = \neg\rho_1(R)$ (resp. $\rho_2(R^\neg) = \neg\rho_2(R)$). It's obvious that $\rho_2(R^\neg) \subseteq \rho_1(R^\neg)$. According to Lemma D.2, we can see that (ρ_1, σ) $\underline{\mathtt{sat}}$ cl'_i implies (ρ_2, σ) $\underline{\mathtt{sat}}$ cl'_i. Therefore, (ρ_1, σ) $\underline{\mathtt{sat}}$ cl_i implies (ρ_2, σ) $\underline{\mathtt{sat}}$ cl_i. □

Lemma D.4 *Let $\rho = \varrho_0, ..., \varrho_n$ be an interpretation, $cls = cl_1, ..., cl_n$ a weakly stratified clause sequence and $1 \leq i \leq n$. We have following two properties:*

Property 1:

- *The set $\{\varrho'_n \mid (\rho[\varrho'_n/\varrho_n], \sigma)$ $\underline{\mathtt{sat}}$ $cl_n\}$ is a Moore Family.*
- *The set $\{\varrho'_i \mid \exists \varrho'_{i+1}, ..., \varrho'_n : (\rho[\varrho'_i/\varrho_i, ..., \varrho'_n/\varrho_n], \sigma)$ $\underline{\mathtt{sat}}$ $cl_i \wedge$ $(\rho[\varrho'_i/\varrho_i, ..., \varrho'_n/\varrho_n], \sigma)$ $\underline{\mathtt{sat}}$ $\mathbf{LFP}(cl_{i+1}, ..., cl_n)\}$ is a Moore Family.*

Property 2:
Assume that $\rho_1 = \varrho^1_0, ..., \varrho^1_n$ and $\rho_2 = \varrho^2_0, ..., \varrho^2_n$ are two interpretations such that 1) $\exists 1 \leq j < i : \varrho^1_j \subseteq \varrho^2_j$, 2) $\forall 0 \leq k < i,\ k \neq j : \varrho^1_k = \varrho^2_k$, and 3) (ρ_l, σ) $\underline{\mathtt{sat}}$ $\mathbf{LFP}(cl_i, ..., cl_n)$ where $l = 1, 2$. Then, we have the following, where $i \leq m$:

- *if cl_m depends positively on cl_j, then $\varrho^1_m \subseteq \varrho^2_m$;*
- *if cl_m depends negatively on cl_j, then $\varrho^1_m \supseteq \varrho^2_m$; and*
- *if cl_m does not depend on cl_j, then $\varrho^1_m = \varrho^2_m$.*

Proof. The Lemma is about two properties of SFP formula $\mathbf{LFP}(cl_i, ..., cl_n)$. We proceed by induction on the number clauses included in $cl_i, ..., cl_n$.

Base case: We first prove the Moore Family property. We consider the set $\{\varrho'_n \mid (\rho[\varrho'_n/\varrho_n], \sigma)$ $\underline{\mathtt{sat}}$ $cl_n\}$. In this case, cl_n is an ALFP formula. From Proposition 2.6, we know that $\{\varrho'_n \mid (\rho[\varrho'_n/\varrho_n], \sigma)$ $\underline{\mathtt{sat}}$ $cl_n\}$ is a Moore Family.

We now consider the property 2. In this case, $i = n$. From the Moore Family property for the base case, we know that (ρ_l, σ) $\underline{\mathtt{sat}}$ cl_n where $l = 1, 2$.

Assume that cl_n positively depends on cl_j. Let $\rho_3 = \varrho^3_0, ..., \varrho^3_n$ be an interpretation such that $\forall 0 \leq i' \leq n-1 : \varrho^3_{i'} = \varrho^1_{i'}$ and $\varrho^3_n = \varrho^2_n$. Since

(ρ_2,σ) $\underline{\texttt{sat}}$ cl_n, according to Lemma D.3, we know that (ρ_3,σ) $\underline{\texttt{sat}}$ cl_n. This means $\varrho_n^2, \varrho_n^3 \in \{\varrho'_n \mid (\rho_1[\varrho'_n/\varrho_n^1],\sigma)\ \underline{\texttt{sat}}\ cl_n\}$. Therefore, we have that $\varrho_n^1 \subseteq \varrho_n^2$.

Assume that cl_n negatively depends on cl_j. Let $\rho_3 = \varrho_0^3, ..., \varrho_n^3$ be an interpretation such that $\forall 0 \leq i' \leq n-1 : \varrho_{i'}^3 = \varrho_{i'}^2$ and $\varrho_n^3 = \varrho_n^1$. Since (ρ_1,σ) $\underline{\texttt{sat}}$ cl_n, according to Lemma D.3, we know that (ρ_3,σ) $\underline{\texttt{sat}}$ cl_n. This means $\varrho_n^1, \varrho_n^3 \in \{\varrho'_n \mid (\rho_2[\varrho'_n/\varrho_n^2],\sigma)\ \underline{\texttt{sat}}\ cl_n\}$. Therefore, we have that $\varrho_n^2 \subseteq \varrho_n^1$.

Assume that cl_n does not depend on cl_j. It's easy to see that $\varrho_n^1 = \varrho_n^2$.

Induction step: We first prove the Moore Family property. Now we consider the set $\Psi = \{\varrho'_i \mid \exists \varrho'_{i+1}, ..., \varrho'_n : (\rho[\varrho'_i/\varrho_i, ..., \varrho'_n/\varrho_n],\sigma)\ \underline{\texttt{sat}}\ cl_i \wedge$ $(\rho[\varrho'_i/\varrho_i, ..., \varrho'_n/\varrho_n],\sigma)\ \underline{\texttt{sat}}\ \mathbf{LFP}(cl_{i+1}, ..., cl_n)\}$. According to the definition of a Moore Family, we need to prove that $\forall \Psi' \in \Psi : \sqcap\Psi' \in \Psi$.

Let Ψ' be an subset of Ψ. We define a set $\Gamma = \{\rho_a | \rho_a = \varrho_0^a, ..., \varrho_n^a,\ \forall 0 \leq i' < i : \varrho_{i'}^a = \varrho_{i'},\ \varrho_i^a \in \Psi',\ (\rho_a,\sigma)\ \underline{\texttt{sat}}\ cl_i,\ (\rho^a,\sigma)\ \underline{\texttt{sat}}\ \mathbf{LFP}(cl_{i+1}, ..., cl_n)\}$. If ρ_a is an interpretation such that $\rho_a \in \Gamma$, then there is an element $\psi' \in \Psi'$ such that $\varrho_i^a = \psi'$. One the other hand, given any element $\psi' \in \Psi'$, we can construct an interpretation ρ_a such that $\varrho_i^a = \psi'$ and $\rho_a \in \Gamma$. Let $\rho_\sqcap = \varrho_0^\sqcap, ..., \varrho_n^\sqcap$ be an interpretation such that $\forall 0 \leq i' < i : \varrho_{i'}^\sqcap = \varrho_{i'},\ \varrho_i^\sqcap = \sqcap\Psi'$ and that $(\rho_\sqcap,\sigma)$ $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_{i+1}, ..., cl_n)$. In the following, we will prove that $(\rho_\sqcap,\sigma)$ $\underline{\texttt{sat}}$ cl_i, since this implies $\sqcap\Psi' \in \Psi$.

Let $\rho_a \in \Gamma$. We compare $\varrho_m^\sqcap$ and ϱ_m^a where $i < m$ in the following.

Assume that cl_i depends positively on cl_m. In this case, it's not possible that cl_m depends negatively on cl_i duo to weak stratification. If cl_m depends positively on cl_i, then, according to the induction hypothesis of property 2, we know that $\varrho_m^\sqcap \subseteq \varrho_m^a$. If cl_m does not depend on cl_i, then, according to the induction hypothesis of property 2, we know that $\varrho_m^\sqcap = \varrho_m^a$. Therefore, $\varrho_m^\sqcap \subseteq \varrho_m^a$ if cl_i depends positively on cl_m.

Assume that cl_i depends negatively on cl_m. In this case, it's not possible that cl_m depends positively on cl_i duo to weak stratification. If cl_m depends negatively on cl_i, then, according to the induction hypothesis of property 2, we know that $\varrho_m^\sqcap \supseteq \varrho_m^a$. If cl_m does not depend on cl_i, then, according to the induction hypothesis of property 2, we know that $\varrho^{\sqcap}_m = \varrho^a_m$. Therefore, $\varrho^{\sqcap}_m \supseteq \varrho^a_m$ if cl_i depends negatively on cl_m.

We define that $\Gamma' = \{\rho_{a'} | \rho_{a'} = \rho_a[\varrho^{\sqcap}_{i+1}/\varrho^a_{i+1}, ..., \varrho^{\sqcap}_n/\varrho^a_n],\ \rho_a \in \Gamma\}$. Let $\rho_a \in \Gamma$ and $\rho_{a'} \in \Gamma'$ such that $\rho_{a'} = \rho_a[\varrho^{\sqcap}_{i+1}/\varrho^a_{i+1}, ..., \varrho^{\sqcap}_n/\varrho^a_n]$. Since (ρ_a, σ) $\underline{\texttt{sat}}$ cl_i, from above and according to Lemma D.3, we know that $(\rho_{a'}, \sigma)$ $\underline{\texttt{sat}}$ cl_i. Since cl_i is an ALFP formula, from Proposition 2.6, we can easily show that $(\rho_{\sqcap}, \sigma)$ $\underline{\texttt{sat}}$ cl_i. This finishes the proof for property 1.

We now consider property 2. Remember that we have proved property 1 above so that we can now use this property. We have three following cases:

Case 1: Assume that cl_i depends positively on cl_j. Let $\rho' = \varrho'_0, ..., \varrho'_n$ be an interpretation such that 1) $\forall 0 \leq k < i : \varrho'_k = \varrho^1_k$, 2) $\varrho'_i = \varrho^2_i$, and 3) (ρ', σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_{i+1}, ..., cl_n)$.

We first compare ϱ^1_i and ϱ^2_i.

Assume that cl_i depends positively on cl_m where $i < m$. In this case, it's not possible that cl_m depends negatively on cl_j because otherwise cl_i depends also negatively on cl_j, which does not satisfy weak stratification. If cl_m depends positively on cl_j, then, according to the induction hypothesis of property 2, we know that $\varrho'_m \subseteq \varrho^2_m$. If cl_m does not depends on cl_j, then, according to the induction hypothesis of property 2, we know that $\varrho'_m = \varrho^2_m$. Therefore, $\varrho'_m \subseteq \varrho^2_m$ if cl_i depends positively on cl_m.

Assume that cl_i depends negatively on cl_m where $i < m$. In this case, it's not possible that cl_m depends positively on cl_j because otherwise cl_i depends also negatively on cl_j, which does not satisfy weak stratification. If cl_m depends negatively on cl_j, then, according to the induction hypothesis of property 2, we know that $\varrho'_m \supseteq \varrho^2_m$. If cl_m does not depends on cl_j, then, according to the induction hypothesis of property 2, we know that $\varrho'_m = \varrho^2_m$. Therefore, $\varrho'_m \supseteq \varrho^2_m$ if cl_i depends negatively on cl_m.

Due to the Moore Family property we have proved above, we have (ρ_2, σ) $\underline{\texttt{sat}}$ cl_i. From above and according to Lemma D.3, we know that (ρ', σ) $\underline{\texttt{sat}}$ cl_i. Let $\Omega_1 = \{\varrho'_i \mid \exists \varrho'_{i+1}, ..., \varrho'_n : (\rho_1[\varrho'_i/\varrho^1_i, ..., \varrho'_n/\varrho^1_n], \sigma)$ $\underline{\texttt{sat}}$ $cl_i \wedge (\rho_1[\varrho'_i/\varrho^1_i, ..., \varrho'_n/\varrho^1_n], \sigma)$ $\underline{\texttt{sat}}$

$\mathbf{LFP}(cl_{i+1}, ..., cl_n)\}$. Since (ρ_1, σ) <u>`sat`</u> $\mathbf{LFP}(cl_i, ..., cl_n)$, we know that $\varrho_i^1 = \sqcap\Omega_1$. Since we know that $\rho' = \rho_1[\varrho_i'/\varrho_i^1, ..., \varrho_n'/\varrho_n^1]$, we have $\varrho_i' \in \Omega_1$. Therefore, $\varrho_i^1 \subseteq \varrho_i'$.

Since $\varrho_i^2 = \varrho_i'$, from above we know that $\varrho_i^1 \subseteq \varrho_i^2$ if cl_i depends positively on cl_j.

Now we compare ϱ_m^1 and ϱ_m^2 where $i < m$. Notice that (ρ_l, σ) <u>`sat`</u> $\mathbf{LFP}(cl_{i+1}, ..., cl_n)$ where $l = 1, 2$.

Assume that cl_m depends positively on cl_j. Then, $\varrho_m' \subseteq \varrho_m^2$. Moreover, cl_m does not depends negatively on cl_i. If cl_m depends positively on cl_i, then, according to the induction hypothesis of property 2, we know that $\varrho_m' \supseteq \varrho_m^1$. If cl_m does not depend on cl_i, then, according to the induction hypothesis of property 2, we know that $\varrho_m' = \varrho_m^1$. From above, $\varrho_m' \supseteq \varrho_m^1$ if cl_m depends positively on cl_j. Therefore, $\varrho_m^1 \subseteq \varrho_m^2$.

Assume that cl_m depends negatively on cl_j. Then, $\varrho_m' \supseteq \varrho_m^2$. Moreover, cl_m does not depends positively on cl_i. If cl_m depends negatively on cl_i, then, according to the induction hypothesis of property 2, we know that $\varrho_m' \subseteq \varrho_m^1$. If cl_m does not depend on cl_i, then, according to the induction hypothesis of property 2, we know that $\varrho_m' = \varrho_m^1$. From above, $\varrho_m' \subseteq \varrho_m^1$ if cl_m depends negatively on cl_j. Therefore, $\varrho_m^1 \supseteq \varrho_m^2$.

Assume that cl_m does not depend on cl_j. Then, according to the induction hypothesis of property 2, $\varrho_m' = \varrho_m^2$. Moreover, cl_m does not depend on cl_i. Therefore, according to the induction hypothesis of property 2, $\varrho_m' = \varrho_m^1$. Therefore, $\varrho_m^1 = \varrho_m^2$.

Therefore, property 2 holds when cl_i depends positively on cl_j.

Case 2: Assume that cl_i depends negatively on cl_j. Let $\rho' = \varrho_0', ..., \varrho_n'$ be an interpretation such that 1) $\forall 0 \leq k < i : \varrho_k' = \varrho_k^2$, 2) $\varrho_i' = \varrho_i^1$, and 3) (ρ', σ) <u>`sat`</u> $\mathbf{LFP}(cl_{i+1}, ..., cl_n)$.

We first compare ϱ_i^1 and ϱ_i^2.

Assume that cl_i depends positively on cl_m where $i < m$. In this case, it's not possible that cl_m depends positively on cl_j because otherwise cl_i depends also positively on cl_j, which does not satisfy weak stratification. If cl_m depends negatively on cl_j, then, according to the induction hypothesis of property 2, we know that $\varrho'_m \subseteq \varrho^1_m$. If cl_m does not depends on cl_j, then, according to the induction hypothesis of property 2, we know that $\varrho'_m = \varrho^1_m$. Therefore, $\varrho'_m \subseteq \varrho^1_m$ if cl_i depends positively on cl_m.

Assume that cl_i depends negatively on cl_m where $i < m$. In this case, it's not possible that cl_m depends negatively on cl_j because otherwise cl_i depends also positively on cl_j, which does not satisfy weak stratification. If cl_m depends positively on cl_j, then, according to the induction hypothesis of property 2, we know that $\varrho'_m \supseteq \varrho^1_m$. If cl_m does not depends on cl_j, then, according to the induction hypothesis of property 2, we know that $\varrho'_m = \varrho^1_m$. Therefore, $\varrho'_m \supseteq \varrho^1_m$ if cl_i depends negatively on cl_m.

Due to the Moore Family property we have proved above, we have $(\rho_1, \sigma)\ \underline{\texttt{sat}}\ cl_i$. From above and according to Lemma D.3, we know that $(\rho', \sigma)\ \underline{\texttt{sat}}\ cl_i$. Let $\Omega_2 = \{\varrho'_i \mid \exists \varrho'_{i+1}, ..., \varrho'_n : (\rho_2[\varrho'_i/\varrho^2_i, ..., \varrho'_n/\varrho^2_n], \sigma)\ \underline{\texttt{sat}}\ cl_i \wedge (\rho_2[\varrho'_i/\varrho^2_i, ..., \varrho'_n/\varrho^2_n], \sigma)\ \underline{\texttt{sat}}\ \mathbf{LFP}(cl_{i+1}, ..., cl_n)\}$. Since $(\rho_2, \sigma)\ \underline{\texttt{sat}}\ \mathbf{LFP}(cl_i, ..., cl_n)$, we know that $\varrho^2_i = \sqcap \Omega_2$. Since we know that $\rho' = \rho_2[\varrho'_i/\varrho^2_i, ..., \varrho'_n/\varrho^2_n]$, we have $\varrho'_i \in \Omega_2$. Therefore, $\varrho^2_i \subseteq \varrho'_i$.

Since $\varrho^1_i = \varrho'_i$, from above we know that $\varrho^2_i \subseteq \varrho^1_i$ if cl_i depends negatively on cl_j.

Now we compare ϱ^1_m and ϱ^2_m where $i < m$. Notice that $(\rho_l, \sigma)\ \underline{\texttt{sat}}$ $\mathbf{LFP}(cl_{i+1}, ..., cl_n)$ where $l = 1, 2$.

Assume that cl_m depends positively on cl_j. Then, $\varrho'_m \supseteq \varrho^1_m$. Moreover, cl_m does not depends positively on cl_i. If cl_m depends negatively on cl_i, then, according to the induction hypothesis of property 2, we know that $\varrho'_m \subseteq \varrho^2_m$. If cl_m does not depend on cl_i, then, according to the induction hypothesis of property 2, we know that $\varrho'_m = \varrho^2_m$. From above, $\varrho'_m \subseteq \varrho^2_m$ if cl_m depends positively on cl_j. Therefore, $\varrho^1_m \subseteq \varrho^2_m$.

Assume that cl_m depends negatively on cl_j. Then, $\varrho'_m \subseteq \varrho^1_m$. Moreover, cl_m does not depends negatively on cl_i. If cl_m depends positively on cl_i, then, according to the induction hypothesis of property 2, we know that $\varrho'_m \supseteq \varrho^2_m$. If cl_m does not depend on cl_i, then, according to the induction hypothesis of property 2, we know that $\varrho'_m = \varrho^2_m$. From above, $\varrho'_m \supseteq \varrho^2_m$ if cl_m depends negatively on cl_j. Therefore, $\varrho^1_m \supseteq \varrho^2_m$.

Assume that cl_m does not depend on cl_j. Then, according to the induction hypothesis of property 2, $\varrho'_m = \varrho^1_m$. Moreover, cl_m does not depend on cl_i. Therefore, according to the induction hypothesis of property 2, $\varrho'_m = \varrho^2_m$. Therefore, $\varrho^1_m = \varrho^2_m$.

Therefore, property 2 holds when cl_i depends negatively on cl_j.

Case 3: Assume that cl_i does not depend on cl_j.

We first compare ϱ^1_i and ϱ^2_i.

Let $\rho' = \varrho'_0, ..., \varrho'_n$ be an interpretation such that 1) $\forall 0 \leq k < i : \varrho'_k = \varrho^1_k$, 2) $\varrho'_i = \varrho^2_i$, and 3) (ρ', σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_{i+1}, ..., cl_n)$.

Assume that cl_i depends on cl_m where $i < m$. In this case, it's not possible that cl_m depends on cl_j because otherwise cl_i depends also on cl_j, which does not satisfy weak stratification. Therefore, according to the induction hypothesis of property 2, we know that $\varrho'_m = \varrho^2_m$.

Due to the Moore Family property we have proved above, we have (ρ_2, σ) $\underline{\texttt{sat}}$ cl_i. From above and according to Lemma D.3, we know that (ρ', σ) $\underline{\texttt{sat}}$ cl_i. Let $\Omega_1 = \{\varrho'_i \mid \exists \varrho'_{i+1}, ..., \varrho'_n : (\rho_1[\varrho'_i/\varrho^1_i, ..., \varrho'_n/\varrho^1_n], \sigma)$ $\underline{\texttt{sat}}$ $cl_i \wedge (\rho_1[\varrho'_i/\varrho^1_i, ..., \varrho'_n/\varrho^1_n], \sigma)$ $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_{i+1}, ..., cl_n)\}$. Since (ρ_1, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_i, ..., cl_n)$, we know that $\varrho^1_i = \sqcap\Omega_1$. Since we know that $\rho' = \rho_1[\varrho'_i/\varrho^1_i, ..., \varrho'_n/\varrho^1_n]$, we have $\varrho'_i \in \Omega_1$. Therefore, $\varrho^1_i \subseteq \varrho'_i$.

Since $\varrho^2_i = \varrho'_i$, from above we know that $\varrho^1_i \subseteq \varrho^2_i$ if cl_i does not depend on cl_j.

On the other hand, let $\rho'' = \varrho''_0, ..., \varrho''_n$ be an interpretation such that 1) $\forall 0 \leq k < i : \varrho''_k = \varrho^2_k$, 2) $\varrho''_i = \varrho^1_i$, and 3) (ρ'', σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_{i+1}, ..., cl_n)$.

Assume that cl_i depends on cl_m where $i < m$. In this case, it's not possible that cl_m depends on cl_j because otherwise cl_i depends also on cl_j, which does not satisfy weak stratification. Therefore, according to the induction hypothesis of property 2, we know that $\varrho''_m = \varrho^1_m$.

Due to the Moore Family property we have proved above, we have (ρ_1, σ) $\underline{\texttt{sat}}$ cl_i. From above and according to Lemma D.3, we know that (ρ'', σ) $\underline{\texttt{sat}}$ cl_i. Let $\Omega_2 = \{\varrho'_i \mid \exists \varrho'_{i+1}, ..., \varrho'_n : (\rho_2[\varrho'_i/\varrho^2_i, ..., \varrho'_n/\varrho^2_n], \sigma)\ \underline{\texttt{sat}}\ cl_i \wedge (\rho_2[\varrho'_i/\varrho^2_i, ..., \varrho'_n/\varrho^2_n], \sigma)$ $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_{i+1}, ..., cl_n)\}$. Since (ρ_2, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_i, ..., cl_n)$, we know that $\varrho^2_i = \sqcap \Omega_2$. Since we know that $\rho'' = \rho_2[\varrho''_i/\varrho^2_i, ..., \varrho''_n/\varrho^2_n]$, we have $\varrho''_i \in \Omega_2$. Therefore, $\varrho^2_i \subseteq \varrho''_i$.

Since $\varrho^1_i = \varrho''_i$, from above we know that $\varrho^2_i \subseteq \varrho^1_i$ if cl_i does not depend on cl_j.

From above, we know that $\varrho^2_i = \varrho^1_i$ if cl_i does not depend on cl_j.

Now we compare ϱ^1_m and ϱ^2_m where $i < m$. Notice that (ρ_l, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_{i+1}, ..., cl_n)$ where $l = 1, 2$.

Assume that cl_m depends positively on cl_j. Then, according to the induction hypothesis of property 2, we know that $\varrho^1_m \subseteq \varrho^2_m$. Assume that cl_m depends negatively on cl_j. Then, according to the induction hypothesis of property 2, we know that $\varrho^1_m \supseteq \varrho^2_m$. Assume that cl_m does not depend on cl_j. Then, according to the induction hypothesis of property 2, $\varrho^1_m = \varrho^2_m$.

Therefore, property 2 holds when cl_i does not depend on cl_j.

Therefore, from the above three cases, we know that property 2 holds. $\square$

Theorem 6.5 Let $\rho = \varrho_0, ..., \varrho_n$ be an interpretation, $cls = cl_1, ..., cl_n$ a weakly stratified clause sequence and $1 \leq i \leq n$. Then, we have the followings:

- The set of interpretations $\{\varrho'_n \mid (\rho[\varrho'_n/\varrho_n], \sigma)\ \underline{\texttt{sat}}\ cl_n\}$ is a Moore Family

- The set of interpretations $\{\varrho'_i \mid \exists \varrho'_{i+1}, ..., \varrho'_n : (\rho[\varrho'_i/\varrho_i, ..., \varrho'_n/\varrho_n], \sigma)\ \underline{\texttt{sat}}\ cl_i \wedge$ $(\rho[\varrho'_i/\varrho_i, ..., \varrho'_n/\varrho_n], \sigma)\ \underline{\texttt{sat}}\ \mathbf{LFP}(cl_{i+1}, ..., cl_n)\}$ is a Moore Family.

PROOF. This is obvious from Lemma D.4. □

Given a clause sequence $cls = cl_1, ..., cl_n$, we can attach *sign* information to it and write cls in the form $cls = cl_1^{\kappa_1}, ..., cl_n^{\kappa_n}$, where $\kappa_i \in \{+, -\}$ for $1 \leq i \leq n$. This introduces the notion of *syntactic monotonicity* defined as follows:

DEFINITION D.5 (SYNTACTIC MONOTONICITY) A clause sequence $cls = cl_1^{\kappa_1}, ..., cl_n^{\kappa_n}$, where $\kappa_i \in \{+, -\}$ for $1 \leq i \leq n$ is *syntactic monotone* if there exists a sign mapping function $\texttt{sign} : \mathcal{R} \rightarrow \{+, -\}$ such that for all relations R defined in cls the following conditions hold:

- if R is defined in $cl_i^{\kappa_i}$, then R is not defined in $cl_j^{\kappa_j}$ and $sign(R) = \kappa_i$;
- if $cl_i^{\kappa_i}$ contains a positive query of R, then $sign(R) = \kappa_i$; and
- if $cl_i^{\kappa_i}$ contains a negative query of R, then $sign(R) \neq \kappa_i$.

LEMMA D.6 *Let $cls = cl_1^{\kappa_1}, ..., cl_n^{\kappa_n}$ be a syntactic monotonic clause sequence and DG_{cls} be its dependency graph. If $cl_j^{\kappa_j}$ depends positively (resp. negatively) on $cl_i^{\kappa_i}$ $(1 \leq i, j \leq n)$, then we have $\kappa_i = \kappa_j$ (resp. $\kappa_i \neq \kappa_j$).*

PROOF. We denote a path from $cl_i^{\kappa_i}$ to $cl_j^{\kappa_j}$ in DG_{cls} by $\pi_{ij} = cl_i^{\kappa_i} \xrightarrow{e_{i,k}} cl_k^{\kappa_k} ... cl_j^{\kappa_j}$ where $1 \leq k \leq n$ and $e_{i,k}$ is the sign labeled to the edge from $cl_i^{\kappa_i}$ to $cl_k^{\kappa_k}$. We define the length of a path to be the number of edges on this path.

Assume that $cl_j^{\kappa_j}$ depends positively (resp. negatively) on $cl_i^{\kappa_i}$. Then, there exists a path π_{ij} such that there are even (resp. odd) number of negative edges on it. We prove by induction on the length of π_{ij}.

Base case: In this case, the path $\pi_{ij} = cl_i^{\kappa_i} \xrightarrow{e_{i,j}} cl_j^{\kappa_j}$ is of length 1. We prove by contradiction.

Assume that $cl_j^{\kappa_j}$ depends positively on $cl_i^{\kappa_i}$ and that $\kappa_i \neq \kappa_j$. In this case, the edge $\xrightarrow{e_{i,j}}$ is a positive edge and we know that a relation defined in $cl_i^{\kappa_i}$ is positively queried in $cl_j^{\kappa_j}$. According to the definition of syntactic monotonicity, we know that $\kappa_i = \kappa_j$. This is a contradiction. Therefore, if $cl_j^{\kappa_j}$ depends positively on $cl_i^{\kappa_i}$, then $\kappa_i = \kappa_j$.

Assume that $cl_j^{\kappa_j}$ depends negatively on $cl_i^{\kappa_i}$ and that $\kappa_i = \kappa_j$. In this case, the edge $\xrightarrow{e_{i,j}}$ is a negative edge and we know that a relation defined in $cl_i^{\kappa_i}$ is negatively queried in $cl_j^{\kappa_j}$. According to the definition of syntactic monotonicity, we know that $\kappa_i \neq \kappa_j$. This is a contradiction. Therefore, $cl_j^{\kappa_j}$ depends negatively on $cl_i^{\kappa_i}$, then $\kappa_i \neq \kappa_j$.

Induction: We denote the sub-path of the path $\pi_{ij} = cl_i^{\kappa_i} \xrightarrow{e_{i,k}} cl_k^{\kappa_k} ... cl_j^{\kappa_j}$ starting from $cl_k^{\kappa_k}$ to $cl_j^{\kappa_j}$ as $\pi_{k,j}$.

Assume that $cl_j^{\kappa_j}$ depends positively on $cl_i^{\kappa_i}$. In this case, there are even number of negations on $\pi_{i,j}$.

If there are even number of negations on $\pi_{k,j}$, we know that $cl_j^{\kappa_j}$ depends positively on $cl_k^{\kappa_k}$. According to the induction hypothesis, we know that $\kappa_k = \kappa_j$. Then, we know that $\xrightarrow{e_{i,k}}$ is a positive edge which means $cl_k^{\kappa_k}$ depends positively on $cl_i^{\kappa_i}$. According to the induction hypothesis, we know that $\kappa_k = \kappa_i$. Therefore, $\kappa_i = \kappa_j$.

If there are odd number of negations on $\pi_{k,j}$, we know that $cl_j^{\kappa_j}$ depends negatively on $cl_k^{\kappa_k}$. According to the induction hypothesis, we know that $\kappa_k \neq \kappa_j$. Then, we know that $\xrightarrow{e_{i,k}}$ is a negative which means $cl_k^{\kappa_k}$ depends negatively on $cl_i^{\kappa_i}$. According to the induction hypothesis, we know that $\kappa_k \neq \kappa_i$. Therefore, $\kappa_i = \kappa_j$.

Assume that $cl_j^{\kappa_j}$ depends negatively on $cl_i^{\kappa_i}$. In this case, there are odd number of negations on $\pi_{i,j}$.

If there are even number of negations on $\pi_{k,j}$, we know that $cl_j^{\kappa_j}$ depends positively on $cl_k^{\kappa_k}$. According to the induction hypothesis, we know that $\kappa_k = \kappa_j$. Then, we know that $\xrightarrow{e_{i,k}}$ is a negative edge which means $cl_k^{\kappa_k}$ depends negatively on $cl_i^{\kappa_i}$. According to the induction hypothesis, we know that $\kappa_k \neq \kappa_i$. Therefore, $\kappa_i \neq \kappa_j$.

If there are odd number of negations on $\pi_{k,j}$, we know that $cl_j^{\kappa_j}$ depends negatively on $cl_k^{\kappa_k}$. According to the induction hypothesis, we know that $\kappa_k \neq \kappa_j$. Then, we know that $\xrightarrow{e_{i,k}}$ is a positive which means $cl_k^{\kappa_k}$ depends positively on $cl_i^{\kappa_i}$. According to the induction hypothesis, we know that $\kappa_k = \kappa_i$. Therefore, $\kappa_i \neq \kappa_j$. □

Lemma D.7 *A clause sequence $cls = cl_1, ..., cl_n$ is weakly stratified if it is syntactic monotone.*

Proof. Assume that cls is syntactic monotone. We write it in the form $cls = cl_1^{\kappa_1}, ..., cl_n^{\kappa_n}$. There are three conditions listed in the definition of weak stratification. It is easy to see that the condition that "if R is defined in cl_i, then R is not defined in cl_j" is satisfied since this is implied by the condition that "if R is defined in $cl_i^{\kappa_i}$, then R is not defined in $cl_j^{\kappa_i}$" in the definition of syntactic monotonicity.

The condition that "cl_i does not depend negatively on itself" should also be satisfied due to Lemma D.6. We prove by contradiction. Assume that cl_i depends negatively on itself. According to Lemma D.6, we know that $\kappa_i \neq \kappa_i$, which is obviously not possible. Therefore, the condition that "cl_i does not depend negatively on itself" is also satisfied.

We will prove by contradiction to show that the last condition in weak stratification is also satisfied. Assume that cl_i depends both positively and negatively on cl_j. According to Lemma D.6, we know that both $\kappa_i = \kappa_i$ and $\kappa_i \neq \kappa_i$ hold, which is not possible. Therefore, the last condition is also satisfied.

From above, we have proved that syntactic monotonicity implies weak stratification. □

Lemma 6.9 Given a closed μ-calculus formula ϕ, assume that $\phi \longmapsto \langle cls_\phi,\ pre_\phi \rangle$ holds according to Table 6.3, the clause sequence cls_ϕ is closed and weakly stratified.

Proof. Let $cls_\phi = cl_1, ..., cl_n$. Remember that we only define one relation in each clause cl_i $(1 \leq i \leq n)$. We will first show that cls_ϕ is syntactic monotone. We pay attention to all the negations in ϕ. For a μ-subformula $\mu Q.\varphi$ in ϕ, we assume that R_Q is defined in cl_i. We require that $\kappa_i = +$ (resp. $\kappa_i = -$) iff $\mu Q.\varphi$ is under an even (resp. odd) number of negations. It's easy to see that $cls_\phi = cl_1^{\kappa_1}, ..., cl_n^{\kappa_n}$ is syntactic monotone. According to Lemma D.7, we know that $cls_\phi = cl_1, ..., cl_n$ is weakly stratified. □

Lemma D.8 *Given a weakly stratified clause sequence $cls = cls_1, cls_2$, where $cls_1 = cl_1, ..., cl_{n_1}$ and $cls_2 = cl_{n_1}, ..., cl_{n_2}$. Assume that no relations defined in cls_1 occur in cls_2 and no relations defined in cls_2 occur in cls_1. Let $\rho = \varrho_0, ..., \varrho_{n_2}$ be an interpretation. Then, (ρ, σ) $\underline{\texttt{sat}}$ $\textbf{LFP}(cls_1, cls_2)$ iff both (ρ, σ) $\underline{\texttt{sat}}$ $\textbf{LFP}(cls_1)$ and (ρ, σ) $\underline{\texttt{sat}}$ $\textbf{LFP}(cls_2)$.*

Proof. We proceed by induction on the number n_1.

Base case $n_1 = 1$: Assume that (ρ, σ) $\underline{\texttt{sat}}$ $\textbf{LFP}(cl_1, cls_2)$ holds. According to the semantics of SFP, we know that (ρ, σ) $\underline{\texttt{sat}}$ $\textbf{LFP}(cls_2)$ and $\varrho_1 = \sqcap\{\varrho_1' \mid \exists \varrho_2', ..., \varrho_{n_2}' : (\rho[\varrho_1'/\varrho_1, ..., \varrho_{n_2}'/\varrho_{n_2}], \sigma)\ \underline{\texttt{sat}} cl_1 \wedge (\rho[\varrho_2'/\varrho_2, ..., \varrho_{n_2}'/\varrho_{n_2}], \sigma)\ \underline{\texttt{sat}}$ $\textbf{LFP}(cls_2)\}$. Since no relations defined in cls_2 occur in cl_1 and, according to Theorem 6.7, we can prove that $\exists \varrho_2', ..., \varrho_{n_2}' : (\rho[\varrho_1'/\varrho_1, ..., \varrho_{n_2}'/\varrho_{n_2}], \sigma)$ $\underline{\texttt{sat}}$ $\textbf{LFP}(cls_2)$ always holds when ϱ_1' is given, we know that $\varrho_1 = \sqcap\{\varrho_1' \mid (\rho[\varrho_1'/\varrho_1], \sigma)$ $\underline{\texttt{sat}} cl_1\}$. Therefore, (ρ, σ) $\underline{\texttt{sat}}$ $\textbf{LFP}(cl_1)$. This finishes the proof of one direction.

Assume that (ρ, σ) $\underline{\texttt{sat}}$ $\textbf{LFP}(cl_1)$ and (ρ, σ) $\underline{\texttt{sat}}$ $\textbf{LFP}(cls_2)$. From (ρ, σ) $\underline{\texttt{sat}}$ $\textbf{LFP}(cl_1)$, we know that $\varrho_1 = \sqcap\{\varrho_1' \mid (\rho[\varrho_1'/\varrho_1], \sigma)\ \underline{\texttt{sat}}\ cl_1\}$. Since no relations defined in cls_2 occur in cl_1 and , according to Theorem 6.7, we can prove that $\exists \varrho_2', ..., \varrho_{n_2}' : (\rho[\varrho_1'/\varrho_1, ..., \varrho_{n_2}'/\varrho_{n_2}], \sigma)$ $\underline{\texttt{sat}}$ $\textbf{LFP}(cls_2)$ always holds when ϱ_1' is given, we know that $\varrho_1 = \sqcap\{\varrho_1' \mid \exists \varrho_2', ..., \varrho_{n_2}' : (\rho[\varrho_1'/\varrho_1, ..., \varrho_{n_2}'/\varrho_{n_2}], \sigma)\ \underline{\texttt{sat}}\ cl_1 \wedge (\rho[\varrho_1'/\varrho_1, ..., \varrho_{n_2}'/\varrho_{n_2}], \sigma)\ \underline{\texttt{sat}}$ $\textbf{LFP}(cls_2)\}$. Since (ρ, σ) $\underline{\texttt{sat}}$ $\textbf{LFP}(cls_2)$ holds, we know that (ρ, σ) $\underline{\texttt{sat}}$ $\textbf{LFP}(cl_1, cls_2)$ holds. This finishes the proof of the other direction.

Induction $n_1 = k + 1$: Assume that (ρ, σ) $\underline{\texttt{sat}}$ $\textbf{LFP}(cls_1, cls_2)$ holds. According to the semantics of SFP, we know that (ρ, σ) $\underline{\texttt{sat}}$ $\textbf{LFP}(cl_2, ..., cl_{k+1}, cls_2)$. According to the induction hypothesis, we know that both (ρ, σ) $\underline{\texttt{sat}}\textbf{LFP}(cl_2, ..., cl_{k+1})$ and (ρ, σ) $\underline{\texttt{sat}}$ $\textbf{LFP}(cls_2)$ holds.

According to the semantics of SFP, we also have that $\varrho_1 = \sqcap\{\varrho'_1 \mid \exists\varrho'_2, ..., \varrho'_{n_2} : (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{n_2}/\varrho_{n_2}], \sigma)$ $\underline{\texttt{sat}}$ $cl_1 \wedge (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{n_2}/\varrho_{n_2}], \sigma)$ $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_2, ..., cl_{k+1}, cls_2)\}$. According to the induction hypothesis, given $\varrho'_1, ..., \varrho'_{n_2}$, we know that $(\rho[\varrho'_1/\varrho_1, ..., \varrho'_{n_2}/\varrho_{n_2}], \sigma)$ $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_2, ..., cl_{k+1}, cls_2)$ iff both $(\rho[\varrho'_1/\varrho_1, ..., \varrho'_{n_2}/\varrho_{n_2}], \sigma)$ $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_2, ..., cl_{k+1})$ and $(\rho[\varrho'_1/\varrho_1, ..., \varrho'_{n_2}/\varrho_{n_2}], \sigma)$$\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_2)$. Since no relations defined in cls_2 occur in cls_1, we know that $\varrho_1 = \sqcap\{\varrho'_1 \mid \exists\varrho'_2, ..., \varrho'_{k+1} : (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{k+1}/\varrho_{k+1}], \sigma)$ $\underline{\texttt{sat}}$ $cl_1 \wedge (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{k+1}/\varrho_{k+1}], \sigma)$ $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_2, ..., cl_{k+1}) \wedge \exists\varrho'_{k+2}, ..., \varrho'_{n_2} : (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{n_2}/\varrho_{n_2}], \sigma)$$\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_2)\}$. Since we can prove, according to Theorem 6.7, that $\exists\varrho'_{k+2}, ..., \varrho'_{n_2} : (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{n_2}/\varrho_{n_2}], \sigma)$$\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_2)$ always holds when $\varrho'_1, ..., \varrho'_{k+1}$ are given, we know that $\varrho_1 = \sqcap\{\varrho'_1 \mid \exists\varrho'_2, ..., \varrho'_{k+1} : (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{k+1}/\varrho_{k+1}], \sigma)$ $\underline{\texttt{sat}}$ $cl_1 \wedge (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{n_2}/\varrho_{n_2}], \sigma)$$\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_2, ..., cl_{k+1})\}$. Since (ρ, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_2, ..., cl_{k+1})$ holds, according to the semantics of SFP, we know that (ρ, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_1)$. This finishes the proof of one direction.

Assume that (ρ, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_1)$ and (ρ, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_2)$ hold. According to the semantics of SFP, from (ρ, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_1)$, we know that (ρ, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_2, ..., cl_{k+1})$. According to the induction hypothesis, we know that $(\rho, \sigma)$$\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_2, ..., cl_{k+1}, cls_2)$.

According to the semantics of SFP, we know that $\varrho_1 = \sqcap\{\varrho'_1 \mid \exists\varrho'_2, ..., \varrho'_{k+1} : (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{k+1}/\varrho_{k+1}], \sigma)$ $\underline{\texttt{sat}}$ $cl_1 \wedge (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{k+1}/\varrho_{k+1}], \sigma)$$\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_2, ..., cl_{k+1})\}$. Since we can prove , according to Theorem 6.7, that $\exists\varrho'_{k+2}, ..., \varrho'_{n_2} : (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{n_2}/\varrho_{n_2}], \sigma)$$\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_2)$ always holds when $\varrho'_1, ..., \varrho'_{k+1}$ are given, we know that $\varrho_1 = \sqcap\{\varrho'_1 \mid \exists\varrho'_2, ..., \varrho'_{k+1} : (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{k+1}/\varrho_{k+1}], \sigma)$ $\underline{\texttt{sat}}$ $cl_1 \wedge (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{k+1}/\varrho_{k+1}], \sigma)$$\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_2, ..., cl_{k+1}) \wedge \exists\varrho'_{k+2}, ..., \varrho'_{n_2} : (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{n_2}/\varrho_{n_2}], \sigma)$$\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_2)\}$. Since no relations defined in cls_2 occur in cls_1, we know that $\varrho_1 = \sqcap\{\varrho'_1 \mid \exists\varrho'_2, ..., \varrho'_{n_2} : (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{n_2}/\varrho_{n_2}], \sigma)$ $\underline{\texttt{sat}}$ $cl_1 \wedge (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{n_2}/\varrho_{n_2}], \sigma)$$\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_2, ..., cl_{k+1}) \wedge (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{n_2}/\varrho_{n_2}], \sigma)$$\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_2)\}$. According to the induction hypothesis, we know that $\varrho_1 = \sqcap\{\varrho'_1 \mid \exists\varrho'_2, ..., \varrho'_{n_2} : (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{n_2}/\varrho_{n_2}], \sigma)$ $\underline{\texttt{sat}}$ $cl_1 \wedge (\rho[\varrho'_1/\varrho_1, ..., \varrho'_{n_2}/\varrho_{n_2}], \sigma)$$\underline{\texttt{sat}}$ $\mathbf{LFP}(cl_2, ..., cl_{k+1}, cls_2)\}$. Therefore, (ρ, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_1, cls_2)$. This finishes the proof of the other direction. □

Theorem 6.10 Let ϕ be a μ-calculus formula with $Q_1, ..., Q_n$ being all the free variables in it. Assume that $\phi \longmapsto \langle cls_\phi,\ pre_\phi\rangle$. Let $\rho = \varrho_0, ..., \varrho_n$ be an interpretation such that (ρ, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_\phi)$, where $\varrho_0(R_{Q_1}) = S_1, ..., \varrho_0(R_{Q_n}) = S_n$ and ϱ_0 defines P_p and T_a. Then, $s' \in [\![\phi]\!]_{e[Q_1 \mapsto S_1, ..., Q_n \mapsto S_n]}$ iff $(\rho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_ϕ.

PROOF. We proceed the proof by structural induction on ϕ.

Case $\phi = p$: According to the semantics of the μ-calculus, we know that $s' \in [\![p]\!]$ iff $p \in L(s')$. According to Table 6.3, we map p to $\langle \mathbf{true},\ P_p(s)\rangle$. Assume that $\rho = \varrho_0, \varrho_1$ and (ρ, σ) $\underline{\mathtt{sat}}$ $\mathbf{LFP}(\mathbf{true})$. Actually here ϱ_1 does not interpret any relations since no relations are defined in the clause **true**. Since ϱ_0 defines P_p, we know that $s \in \varrho_0(P_p)$ if and only if $p \in L(s)$. Therefore, $(\rho, \sigma[s \mapsto s'])$ $\underline{\mathtt{sat}}$ $P_p(s)$ iff $s' \in \rho(P_p)$ iff $s' \in \varrho_0(P_p)$ iff $p \in L(s')$. This means $s' \in [\![p]\!]$ iff $(\rho, \sigma[s \mapsto s'])$ $\underline{\mathtt{sat}}$ $P_p(s)$.

Case $\phi = Q$: Let $e' = e[Q \mapsto S]$. According to the semantics of the μ-calculus, we know that $s' \in [\![Q]\!]_{e'}$ iff $s' \in e'(Q)$ iff $s' \in S$. According to Table 6.3, we map Q to $\langle \mathbf{true},\ R_Q(s)\rangle$. Assume that $\rho = \varrho_0, \varrho_1$ and (ρ, σ) $\underline{\mathtt{sat}}$ $\mathbf{LFP}(\mathbf{true})$. Here, ϱ_1 does not interpret any relations since no relations are defined in the clause **true**. Since $\varrho_0(R_Q) = S$, we know that $(\rho, \sigma[s \mapsto s'])$ $\underline{\mathtt{sat}}$ $R_Q(s)$ iff $s' \in \rho(R_Q)$ iff $s' \in \varrho_0(R_Q)$ iff $s' \in S$. This means $s' \in [\![Q]\!]_{e'}$ iff $(\rho, \sigma[s \mapsto s'])$ $\underline{\mathtt{sat}}$ $R_Q(s)$.

Case $\phi = \phi_1 \vee \phi_2$: Assume that $Q_1, ..., Q_n$ are all the free variables in $\phi_1 \vee \phi_2$. Let $e' = e[Q_1 \mapsto S_1, ..., Q_n \mapsto S_n]$. According to Table 6.3, we know that $cls_{\phi_1 \vee \phi_2} = cls_{\phi_1}, cls_{\phi_2}$. Assume that (ρ, σ) $\underline{\mathtt{sat}}$ $\mathbf{LFP}(cls_{\phi_1 \vee \phi_2})$.

From $\phi_1 \vee \phi_2$, we know that no bounded variables in ϕ_1 occur in ϕ_2 and no bounded variables in ϕ_2 occur in ϕ_1. Therefore, no relations defined in cl_{ϕ_1} occur in cl_{ϕ_2} and no relations defined in cl_{ϕ_2} occur in cl_{ϕ_1}. From Lemma D.8, we know that (ρ, σ) $\underline{\mathtt{sat}}$ $\mathbf{LFP}(cls_{\phi_1})$ and (ρ, σ) $\underline{\mathtt{sat}}$ $\mathbf{LFP}(cls_{\phi_2})$. According to the induction hypothesis, we know that $s' \in [\![\phi_1]\!]_{e'}$ iff $(\rho, \sigma[s \mapsto s'])$ $\underline{\mathtt{sat}}$ pre_{ϕ_1} and $s' \in [\![\phi_2]\!]_{e'}$ iff $(\rho, \sigma[s \mapsto s'])$ $\underline{\mathtt{sat}}$ pre_{ϕ_2}.

According to the semantics of the μ-calculus, $s' \in [\![\phi_1 \vee \phi_2]\!]_{e'}$ iff $s' \in [\![\phi_1]\!]_{e'}$ or $s' \in [\![\phi_2]\!]_{e'}$ holds. According to the semantics of SFP, we know that $(\rho, \sigma[s \mapsto s'])$ $\underline{\mathtt{sat}}$ $pre_{\phi_1} \vee pre_{\phi_2}$ iff $(\rho, \sigma[s \mapsto s'])$ $\underline{\mathtt{sat}}$ pre_{ϕ_1} or $(\rho, \sigma[s \mapsto s'])$ $\underline{\mathtt{sat}}$ pre_{ϕ_2} holds. Therefore, $s' \in [\![\phi_1 \vee \phi_2]\!]_{e'}$ iff $(\rho, \sigma[s \mapsto s'])$ $\underline{\mathtt{sat}}$ $pre_{\phi_1} \vee pre_{\phi_2}$.

Case $\phi = \phi_1 \wedge \phi_2$: This case is similar to $\phi = \phi_1 \vee \phi_2$.

Case $\phi = \langle a \rangle \varphi$: Assume that $Q_1, ..., Q_n$ are all the free variables in $\langle a \rangle \varphi$. Let $e' = e[Q_1 \mapsto S_1, ..., Q_n \mapsto S_n]$. According to Table 6.3, we know that

$\langle a\rangle\varphi \longmapsto \langle cl_\varphi,\ \exists s' : T_a(s,s') \wedge pre_\varphi[s'/s]\rangle$. Assume that (ρ,σ) <u>sat</u> $\mathbf{LFP}(cls_{\langle a\rangle\varphi})$. Since $cls_{\langle a\rangle\varphi} = cls_\varphi$, we know that (ρ,σ) <u>sat</u> $\mathbf{LFP}(cls_\varphi)$. Therefore, according to the induction hypothesis, $t \in [\![\varphi]\!]_{e'}$ iff $(\rho,\sigma[s\mapsto t])$ <u>sat</u> pre_φ.

According to the semantics of the μ-calculus, $s'' \in [\![\langle a\rangle\varphi]\!]_{e'}$ iff $\exists t : (s'',t) \in a \wedge t \in [\![\varphi]\!]_{e'}$ holds. Notice that $(s'',t) \in \rho(T_a)$ iff $(s'',t) \in a$. According to the semantics of SFP, $(\rho,\sigma[s\mapsto s''])$ <u>sat</u> $\exists s' : T_a(s,s') \wedge pre_\varphi[s'/s]$ iff $(\rho,\sigma[s\mapsto s''][s'\mapsto t])$ <u>sat</u> $T_a(s,s') \wedge pre_\varphi[s'/s]$ holds for some $t \in S$ iff $(s'',t) \in \rho(T_a) \wedge (\rho,\sigma[s\mapsto s''][s'\mapsto t])$ <u>sat</u> $pre_\varphi[s'/s]$ for some $t \in S$. Since $(\rho,\sigma[s\mapsto s''][s'\mapsto t])$ <u>sat</u> $pre_\varphi[s'/s]$ iff $((\rho,\sigma[s\mapsto t])$ <u>sat</u> pre_φ iff $t \in [\![\varphi]\!]_{e'}$. We can see that $s'' \in [\![\langle a\rangle\varphi]\!]_{e'}$ iff $(\rho,\sigma[s\mapsto s''])$ <u>sat</u> $\exists s' : T_a(s,s') \wedge pre_\phi[s'/s]$.

Case $\phi = [a]\varphi$: Assume that $Q_1,...,Q_n$ are all the free variables in $[a]\varphi$. Let $e' = e[Q_1\mapsto S_1,...,Q_n\mapsto S_n]$. According to Table 6.3, we know that $[a]\phi \longmapsto \langle cl_\phi,\ \forall s' : \neg T_a(s,s') \vee pre_\phi[s'/s]\rangle$. Assume that (ρ,σ) <u>sat</u> $\mathbf{LFP}(cls_{[a]\varphi})$. Since $cls_{[a]\varphi} = cls_\varphi$, we know that (ρ,σ) <u>sat</u> $\mathbf{LFP}(cls_\varphi)$. Therefore, according to the induction hypothesis, $t \in [\![\varphi]\!]_{e'}$ iff $(\rho,\sigma[s\mapsto t])$ <u>sat</u> pre_φ.

According to the semantics of the μ-calculus, $s'' \in [\![[a]\varphi]\!]_{e'}$ iff $\forall t : (s'',t) \in a$ implies $t \in [\![\varphi]\!]_{e'}$ iff $\forall t : (s'',t) \notin a \vee t \in [\![\varphi]\!]_{e'}$. Notice that $(s'',t) \notin \rho(T_a)$ iff $(s'',t) \notin a$. According to the semantics of SFP, $(\rho,\sigma[s\mapsto s''])$ <u>sat</u> $\forall s' : \neg T_a(s,s') \vee pre_\phi[s'/s]$ iff $(\rho,\sigma[s\mapsto s''][s'\mapsto t])$ <u>sat</u> $\neg T_a(s,s') \vee pre_\phi[s'/s]$ holds for all $t \in S$ iff $(s'',t) \notin \rho(T_a) \vee (\rho,\sigma[s\mapsto s''][s'\mapsto t])$ <u>sat</u> $pre_\varphi[s'/s]$ for all $t \in S$. Since $(\rho,\sigma[s\mapsto s''][s'\mapsto t])$ <u>sat</u> $pre_\varphi[s'/s]$ iff $((\rho,\sigma[s\mapsto t])$ <u>sat</u> pre_φ iff $t \in [\![\varphi]\!]_{e'}$. We can see that $s'' \in [\![[a]\varphi]\!]_{e'}$ iff $(\rho,\sigma[s\mapsto s''])$ <u>sat</u> $\forall s' : \neg T_a(s,s') \vee pre_\phi[s'/s]$.

Case $\phi = \neg\mu Q.\varphi$: Assume that $Q_1,...,Q_n$ are all the free variables in $\neg\mu Q.\varphi$. Let $e' = e[Q_1\mapsto S_1,...,Q_n\mapsto S_n]$. Assume that (ρ,σ) <u>sat</u> $\mathbf{LFP}(cls_{\neg\mu Q.\varphi})$. Since $cls_{\neg\mu Q.\varphi} = cls_{\mu Q.\varphi}$, we know that (ρ,σ) <u>sat</u> $\mathbf{LFP}(cls_{\mu Q.\varphi})$. Therefore, according to the induction hypothesis, $s' \in [\![\mu Q.\varphi]\!]_{e'}$ iff $(\rho,\sigma[s\mapsto s'])$ <u>sat</u> $R_Q(s)$.

According to the semantics of the μ-calculus, $s' \in [\![\neg\mu Q.\varphi]\!]_{e'}$ iff $s' \notin [\![\mu Q.\varphi]\!]_{e'}$. According to the semantics of SFP, $(\rho,\sigma[s\mapsto s'])$ <u>sat</u> $\neg R_Q(s)$ iff $s' \notin \rho(R_Q)$. Since $s' \notin \rho(R_Q)$ iff $(\rho,\sigma[s\mapsto s'])$ <u>sat</u> $R_Q(s)$ does not hold. Therefore, $s' \in [\![\neg\mu Q.\varphi]\!]_{e'}$ iff $(\rho,\sigma[s\mapsto s'])$ <u>sat</u> $\neg R_Q(s)$.

Case $\phi = \mu Q.\varphi$: Assume that $Q_1,...,Q_n$ are all the free variables in $\mu Q.\varphi$. Let $e' = e[Q_1\mapsto S_1,...,Q_n\mapsto S_n]$. According to Table 6.3, we know that $cls_{\mu Q.\varphi} =$

$[\forall s : pre_\phi \Rightarrow R_Q(s)], cls_\varphi$. Assume $\rho = \varrho_1, ..., \varrho_n$ and (ρ, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_{\mu Q.\varphi})$. We write $cls_{\mu Q.\varphi}$ in the form $cls_{\mu Q.\varphi} = cl_1, cls_\varphi$, where $cl_1 = [\forall s : pre_\phi \Rightarrow R_Q(s)]$. According to the semantics of SFP, we know that (ρ, σ) $\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_\varphi)$ and $\varrho_1 = \sqcap\{\varrho'_1 \mid \exists \varrho'_2, ..., \varrho'_n : (\rho[\varrho'_1/\varrho_1, ..., \varrho'_n/\varrho_n], \sigma)$ $\underline{\texttt{sat}}$ $[\forall s : pre_\phi \Rightarrow R_Q(s)] \wedge (\rho[\varrho'_1/\varrho_1, ..., \varrho'_n/\varrho_n], \sigma)$ $\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_\varphi)\}$. Here, ϱ_1 only interpret the relation R_Q.

According to the semantics of SFP, we know that $\varrho_1 = \sqcap\{\varrho'_1 \mid \exists \varrho'_2, ..., \varrho'_n :$ $(\rho[\varrho'_1/\varrho_1, ..., \varrho'_n/\varrho_n], \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $[pre_\phi \Rightarrow R_Q(s)]$ `for all` $s' \in \mathcal{U} \wedge$ $(\rho[\varrho'_1/\varrho_1, ..., \varrho'_n/\varrho_n], \sigma)$ $\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_\varphi)\} = \sqcap\{\varrho'_1 \mid \exists \varrho'_2, ..., \varrho'_n :$ $s' \in \rho[\varrho'_1/\varrho_1, ..., \varrho'_n/\varrho_n](R_Q)$ `whenever` $(\rho[\varrho'_1/\varrho_1, ..., \varrho'_n/\varrho_n], \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_φ `for all` $s' \in \mathcal{U} \wedge (\rho[\varrho'_1/\varrho_1, ..., \varrho'_n/\varrho_n], \sigma)$ $\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_\varphi)\} = \sqcap\{\varrho'_1 \mid \exists \varrho'_2, ..., \varrho'_n :$ $\{s' | (\rho[\varrho'_1/\varrho_1, ..., \varrho'_n/\varrho_n], \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $pre_\varphi\} \subseteq \rho[\varrho'_1/\varrho_1, ..., \varrho'_n/\varrho_n](R_Q) \wedge$ $(\rho[\varrho'_1/\varrho_1, ..., \varrho'_n/\varrho_n], \sigma)$ $\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_\varphi)\}$.

Assume that $(\rho[\varrho'_1/\varrho_1, ..., \varrho'_n/\varrho_n], \sigma)$ $\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_\varphi)$ where $\varrho'_1, ..., \varrho'_n$ are given. According to the induction hypothesis, we know that $s' \in [\![\varphi]\!]_{e'[Q \mapsto \varrho'_1(R_Q)]}$ iff $(\rho[\varrho'_1/\varrho_1, ..., \varrho'_n/\varrho_n], \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ pre_φ. Therefore, we know that $\varrho_1 = \sqcap\{\varrho'_1 \mid$ $\exists \varrho'_2, ..., \varrho'_n :\ \{s' | s' \in [\![\varphi]\!]_{e'[Q \mapsto \varrho'_1(R_Q)]}\} \subseteq \varrho'_1(R_Q) \wedge (\rho[\varrho'_1/\varrho_1, ..., \varrho'_n/\varrho_n], \sigma)$ $\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_\varphi)\} = \sqcap\{\varrho'_1 \mid \{s' | s' \in [\![\varphi]\!]_{e'[Q \mapsto \varrho'_1(R_Q)]}\} \subseteq \varrho'_1(R_Q) \wedge \exists \varrho'_2, ..., \varrho'_n :$ $(\rho[\varrho'_1/\varrho_1, ..., \varrho'_n/\varrho_n], \sigma)$ $\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_\varphi)\}$. Since we can prove, according to Theorem 6.7, that $\exists \varrho'_2, ..., \varrho'_n : (\rho[\varrho'_1/\varrho_1, ..., \varrho'_n/\varrho_n], \sigma)$ $\underline{\texttt{sat}}$ $\mathbf{LFP}(cls_\varphi)$ always holds, we know that $\varrho_1 = \sqcap\{\varrho'_1 \mid \{s' | s' \in [\![\varphi]\!]_{e'[Q \mapsto \varrho'_1(R_Q)]}\} \subseteq \varrho'_1(R_Q)\}$. This exactly mimics the μ-calculus semantics of $\mu Q.\varphi$. Therefore, we know that $s' \in [\![\mu Q.\varphi]\!]_{e'}$ iff $(\rho, \sigma[s \mapsto s'])$ $\underline{\texttt{sat}}$ $R_Q(s)$. $\square$

Bibliography

[1] A. Pnueli. The Temporal Logic of Programs. In *Proc. of the 18th IEEE Symposium on Foundations of Computer Science*, pages 46-77, 1977.

[2] Edmund M. Clarke, Orna Grumberg, and Doron A.Peled: Model Checking. MIT Press, 1999.

[3] Edmund M. Clarke, E. Allen Emerson: Design and Synthesis of Synchronization Skeletons Using Branching-Time Temporal Logic. Logic of Programs 1981: 52-71

[4] E. Allen Emerson, Edmund M. Clarke: Characterizing Correctness Properties of Parallel Programs Using Fixpoints. ICALP 1980: 169-181

[5] Mordechai Ben-Ari, Amir Pnueli, Zohar Manna: The Temporal Logic of Branching Time. Acta Inf. 20: 207-226 (1983)

[6] E. Allen Emerson, Chin-Laung Lei: Efficient Model Checking in Fragments of the Propositional Mu-Calculus (Extended Abstract) *LICS 1986*: 267-278

[7] Rance Cleaveland, Bernhard Steffen: A Linear-Time Model-Checking Algorithm for the Alternation-Free Modal Mu-Calculus. *Formal Methods in System Design 2*(2): 121-147 (1993)

[8] Henrik Reif Andersen: Model Checking and Boolean Graphs. *Theor. Comput. Sci. 126*(1): 3-30 (1994)

[9] Rance Cleaveland, Marion Klein, Bernhard Steffen: Faster Model Checking for the Modal Mu-Calculus. CAV 1992: 410-422

[10] Christel Baier, Joost-Pieter Katoen: Principles of model checking. MIT Press 2008: I-XVII, 1-975

[11] Flemming Nielson, Hanne Riis Nielson, Chris Hankin: Principles of program analysis (2. corr. print). Springer 2005: I-XXI, 1-452

[12] Rance Cleaveland: Tableau-Based Model Checking in the Propositional Mu-Calculus. Acta Inf. 27(8): 725-747 (1989)

[13] Colin Stirling, David Walker: Local Model Checking in the Modal mu-Calculus. Theor. Comput. Sci. 89(1): 161-177 (1991)

[14] Dexter Kozen: Results on the Propositional mu-Calculus. *Theor. Comput. Sci. 27*: 333-354 (1983)

[15] Hanne Riis Nielson, Flemming Nielson: Flow Logic: A Multi-paradigmatic Approach to Static Analysis. *The Essence of Computation 2002*: 223-244

[16] Bernhard Steffen: Data Flow Analysis as Model Checking. *TACS 1991*: 346-365

[17] Bernhard Steffen: Generating Data Flow Analysis Algorithms from Modal Specifications. *Sci. Comput. Program. 21*(2): 115-139 (1993)

[18] Kim Guldstrand Larsen: Efficient Local Correctness Checking. CAV 1992: 30-43

[19] David A. Schmidt, Bernhard Steffen: Program Analysis as Model Checking of Abstract Interpretations. *SAS 1998*: 351-380

[20] David A. Schmidt: Data Flow Analysis is Model Checking of Abstract Interpretations. *POPL 1998*: 38-48

[21] Flemming Nielson, Hanne Riis Nielson: Model Checking Is Static Analysis of Modal Logic. *FOSSACS 2010*: 191-205

[22] Chiara Bodei, Pierpaolo Degano, Flemming Nielson, Hanne Riis Nielson: Flow logic for Dolev-Yao secrecy in cryptographic processes. Future Generation Comp. Syst. 18(6): 747-756 (2002)

[23] Flemming Nielson, Hanne Riis Nielson, Rene Rydhof Hansen: Validating firewalls using flow logics. Theor. Comput. Sci. 283(2): 381-418 (2002)

[24] Chiara Bodei, Pierpaolo Degano, Flemming Nielson, Hanne Riis Nielson: Security Analysis using Flow Logics. Bulletin of the EATCS 70: 112-130 (2000)

[25] Flemming Nielson, Hanne Riis Nielson: Flow Logic for Imperative Objects. MFCS 1998: 220-228

[26] Flemming Nielson, Hanne Riis Nielson: Flow Logic and Operational Semantics. Electr. Notes Theor. Comput. Sci. 10: 150-169 (1997)

[27] Hanne Riis Nielson, Flemming Nielson: Flow Logics for Constraint Based Analysis. CC 1998: 109-127

[28] Rocco De Nicola, Frits W. Vaandrager: Action versus State based Logics for Transition Systems. *Semantics of Systems of Concurrent Processes 1990*: 407-419

[29] Flemming Nielson, Helmut Seidl, Hanne Riis Nielson: A Succinct Solver for ALFP. *Nord. J. Comput. 9*(4): 335-372 (2002)

[30] Flemming Nielson: Two-Level Semantics and Abstract Interpretation. *Theor. Comput. Sci. 69*(2): 117-242 (1989)

[31] Hanne Riis Nielson, Flemming Nielson, Henrik Pilegaard: Flow Logic for Process Calculi. *ACM Comput. Surv.* 44(1): 3 (2012)

[32] Krzysztof R. Apt, Howard A. Blair, Adrian Walker: Towards a Theory of Declarative Knowledge. *Foundations of Deductive Databases and Logic Programming. 1988*: 89-148

[33] Ashok K. Chandra, David Harel: Computable Queries for Relational Data Bases. *J. Comput. Syst. Sci.* 21(2): 156-178 (1980)

[34] Y. S. Ramakrishna, C. R. Ramakrishnan, I. V. Ramakrishnan, Scott A. Smolka, Terrance Swift, David Scott Warren: Efficient Model Checking Using Tabled Resolution. *CAV 1997*: 143-154

[35] C. R. Ramakrishnan: A Model Checker for Value-Passing Mu-Calculus Using Logic Programming. *PADL 2001*: 1-13

[36] C. R. Ramakrishnan, I. V. Ramakrishnan, Scott A. Smolka, Yifei Dong, Xiaoqun Du, Abhik Roychoudhury, V. N. Venkatakrishnan: XMC: A Logic-Programming-Based Verification Toolset. *CAV 2000*: 576-580

[37] Giorgio Delzanno, Andreas Podelski: Model Checking in CLP. *TACAS 1999*: 223-239

[38] Giorgio Delzanno, Andreas Podelski: Constraint-based deductive model checking. *STTT* 3(3): 250-270 (2001)

[39] Michael Leuschel, Thierry Massart: Infinite State Model Checking by Abstract Interpretation and Program Specialisation. LOPSTR 1999: 62-81 (1999)

[40] Patrice Godefroid, Radha Jagadeesan: Automatic Abstraction Using Generalized Model Checking. CAV 2002: 137-150

[41] Sharon Shoham, Orna Grumberg: A game-based framework for CTL counterexamples and 3-valued abstraction-refinement. ACM Trans. Comput. Log. 9(1): (2007)

[42] Marsha Chechik, Steve M. Easterbrook, Victor Petrovykh: Model-Checking over Multi-valued Logics. FME 2001: 72-98

[43] Marsha Chechik, Benet Devereux, Steve M. Easterbrook, Arie Gurfinkel: Multi-valued symbolic model-checking. ACM Trans. Softw. Eng. Methodol. 12(4): 371-408 (2003)

[44] Marsha Chechik, Arie Gurfinkel, Benet Devereux: chi-Chek: A Multi-valued Model-Checker. CAV 2002: 505-509

[45] Beata Konikowska, Wojciech Penczek: On Designated Values in Multi-valued CTL* Model Checking. Fundam. Inform. 60(1-4): 211-224 (2004)

[46] Thomas W. Reps, Shmuel Sagiv, Reinhard Wilhelm: Static Program Analysis via 3-Valued Logic. CAV 2004: 15-30

[47] Shmuel Sagiv, Thomas W. Reps, Reinhard Wilhelm: Parametric shape analysis via 3-valued logic. ACM Trans. Program. Lang. Syst. 24(3): 217-298 (2002)

[48] Stephen Cole Kleene. *Introduction to Metamathematics*. North Holland, 1987.

[49] Glenn Bruns, Patrice Godefroid: Generalized Model Checking: Reasoning about Partial State Spaces. CONCUR 2000: 168-182

[50] Michael Huth, Mark Ryan. *Logic in Computer Science: Modelling and Reasoning about Systems*. Cambridge University Press, 2004

[51] Piotr Filipiuk, Hanne Riis Nielson, Flemming Nielson: Explicit Versus Symbolic Algorithms for Solving ALFP Constraints. *Electr. Notes Theor. Comput. Sci.* 267(2): 15-28 (2010)

[52] Kim Guldstrand Larsen, Bent Thomsen: A Modal Process Logic *LICS 1988*: 203-210

[53] Kim Guldstrand Larsen: Modal Specifications. *Automatic Verification Methods for Finite State Systems 1989*: 232-246

[54] Tal Lev-Ami, Shmuel Sagiv: TVLA: A System for Implementing Static Analyses. *SAS 2000*: 280-301

[55] Tal Lev-Ami, Roman Manevich, Shmuel Sagiv: TVLA: A system for generating abstract interpreters. *IFIP Congress Topical Sessions 2004*: 367-376

[56] Michael Huth, Radha Jagadeesan, David A. Schmidt: Modal Transition Systems: A Foundation for Three-Valued Program Analysis. *ESOP 2001:* 155-169

[57] Igor Bogudlov, Tal Lev-Ami, Thomas W. Reps, Mooly Sagiv: Revamping TVLA: Making Parametric Shape Analysis Competitive. *CAV 2007*: 221-225

[58] Leonard Bolc, Piotr Borowik: *Many-Valued Logics*. Springer Verlag, 1992.

[59] Brian A. Davey, Hilary A. Priestley:*Introduction to Lattices and Order* (2. ed.). Cambridge University Press 2002: I-XII, 1-298

[60] E. Allen Emerson, Chin-Laung Lei: Modalities for Model Checking: Branching Time Strikes Back. POPL 1985: 84-96

[61] Hongyan Sun, Hanne Riis Nielson, Flemming Nielson: Data Structures in the Succinct Solver(V1.0). Technical Report, SECSAFE-IMM-005-1.0, 2002.

[62] Patrick Cousot, Radhia Cousot: Abstract Interpretation: A Unified Lattice Model for Static Analysis of Programs by Construction or Approximation of Fixpoints. POPL 1977: 238-252

[63] Patrick Cousot, Radhia Cousot: Systematic Design of Program Analysis Frameworks. POPL 1979: 269-282

[64] Glenn Bruns, Patrice Godefroid: Model Checking with Multi-valued Logics. ICALP 2004: 281-293

[65] Luca Cardelli, Andrew D. Gordon: Mobile Ambients. FoSSaCS 1998: 140-155

[66] Hanne Riis Nielson, Flemming Nielson, Mikael Buchholtz: Security for Mobility. FOSAD 2002: 207-265

[67] Fuyuan Zhang, Flemming Nielson, Hanne Riis Nielson: Multi-valued Static Analysis. Under Submission.

[68] Fuyuan Zhang, Flemming Nielson, Hanne Riis Nielson: Fixpoints vs Moore Families. Student Research Forum at SOFSEM 2012.

[69] Fuyuan Zhang, Flemming Nielson, Hanne Riis Nielson: Model Checking as Static Analysis: Revisited. IFM 2012: 99-112

[70] M.S.Hecht: Flow Analysis of Computer Programs. North Holland, 1977.

[71] T.J.Marlowe and B.G.Ryder: Properties of Data Flow Frameworks- a Unified Model. Acta Informatica, 28(2):121-163,1990.

[72] Olin Shivers: Control-Flow Analysis in Scheme. PLDI 1988: 164-174

[73] Olin Shivers: The Semantics of Scheme Control-Flow Analysis. PEPM 1991: 190-198

[74] Pierre Jouvelot, David K. Gifford: Reasoning about Continuations with Control Effects. PLDI 1989: 218-226

[75] Pierre Jouvelot, David K. Gifford: Algebraic Reconstruction of Types and Effects. POPL 1991: 303-310

[76] A.V.Aho, J.E.Hopcroft, and J.D.Ullman: The Design and Analysis of Computer Algorithms. Addison Wesley, 1974.

[77] Edmund M. Clarke, Orna Grumberg, Kiyoharu Hamaguchi: Another Look at LTL Model Checking. Formal Methods in System Design 10(1): 47-71 (1997)

[78] Patrice Godefroid, Radha Jagadeesan: On the Expressiveness of 3-Valued Models. VMCAI 2003: 206-222

[79] Edmund M. Clarke: The Birth of Model Checking. 25 Years of Model Checking 2008: 1-26

[80] Park, D.M.R: Finiteness is mu-ineffable. Theory of Computation Report No. 3, Warwich 1974

[81] Tarski, A: A Lattice-theoretical Fixpoint Theorem and Its Application. Pacific J. Math. 5, 285-309 1955

[82] Kleene, S.C: Introduction to Metamathematics, Wolters-Noordhoff, Groningen 1971

[83] Jerry R. Burch, Edmund M. Clarke, Kenneth L. McMillan, David L. Dill, L. J. Hwang: Symbolic Model Checking: 10^{20} States and Beyond LICS 1990: 428-439

[84] Kenneth L. McMillan: Symbolic model checking. Kluwer 1993: I-XV, 1-194

[85] Jerry R. Burch, Edmund M. Clarke, Kenneth L. McMillan, David L. Dill, L. J. Hwang: Symbolic Model Checking: 10^{20} States and Beyond Inf. Comput. 98(2): 142-170 (1992)

[86] Antti Valmari: A Stubborn Attack On State Explosion. CAV 1990: 156-165

[87] Patrice Godefroid: Using Partial Orders to Improve Automatic Verification Methods. CAV 1990: 176-185

[88] Doron Peled: Combining Partial Order Reductions with On-the-fly Model-Checking. CAV 1994: 377-390

[89] Cliff B. Jones: Specification and Design of (Parallel) Programs. IFIP Congress 1983: 321-332

[90] Orna Grumberg, David E. Long: Model Checking and Modular Verification. ACM Trans. Program. Lang. Syst. 16(3): 843-871 (1994)

[91] Jayadev Misra, K. Mani Chandy: Proofs of Networks of Processes. IEEE Trans. Software Eng. 7(4): 417-426 (1981)

[92] A. Pnueli. In Transition from Global to Modular Temporal Reasoning about Programs. In K.R. Apt, editor, Logics and Models of Concurrent Systems, sub-series F: Computer and System Science, pages 123-144. Springer-Verlag, 1985.

[93] Edmund M. Clarke, Orna Grumberg, David E. Long: Model Checking and Abstraction. ACM Trans. Program. Lang. Syst. 16(5): 1512-1542 (1994)

[94] Edmund M. Clarke, Orna Grumberg, David E. Long: Model Checking and Abstraction. POPL 1992: 342-354

[95] Saddek Bensalem, Ahmed Bouajjani, Claire Loiseaux, Joseph Sifakis: Property Preserving Simulations. CAV 1992: 260-273

[96] C. Norris Ip, David L. Dill: Better Verification Through Symmetry. CHDL 1993: 97-111

[97] Edmund M. Clarke, Thomas Filkorn, Somesh Jha: Exploiting Symmetry In Temporal Logic Model Checking. CAV 1993: 450-462

[98] E. Allen Emerson, A. Prasad Sistla: Symmetry and Model Checking. CAV 1993: 463-478

[99] Edmund M. Clarke, Orna Grumberg, Michael C. Browne: Reasoning About Networks With Many Identical Finite-State Processes. PODC 1986: 240-248

[100] Robert P. Kurshan, Kenneth L. McMillan: A Structural Induction Theorem for Processes. PODC 1989: 239-247

[101] Pierre Wolper, Vinciane Lovinfosse: Verifying Properties of Large Sets of Processes with Network Invariants. Automatic Verification Methods for Finite State Systems 1989: 68-80

[102] John B. Kam, Jeffrey D. Ullman: Monotone Data Flow Analysis Frameworks. Acta Inf. 7: 305-317 (1977)

[103] Jens Palsberg: Closure Analysis in Constraint Form. ACM Trans. Program. Lang. Syst. 17(1): 47-62 (1995)

[104] Patrick Cousot, Radhia Cousot: Comparison of the Galois Connection and Widening/Narrowing Approaches to Abstract Interpretation. JTASPEFT/WSA 1991: 107-110

[105] John M. Lucassen, David K. Gifford: Polymorphic Effect Systems. POPL 1988: 47-57

[106] Dennis Dams, Rob Gerth, Orna Grumberg: Abstract Interpretation of Reactive Systems. ACM Trans. Program. Lang. Syst. 19(2): 253-291 (1997)

[107] Witold Charatonik, Andreas Podelski: Set-Based Analysis of Reactive Infinite-State Systems. TACAS 1998: 358-375

[108] Andreas Podelski: Model Checking as Constraint Solving. SAS 2000: 22-37

[109] Patrick Cousot, Radhia Cousot: Temporal Abstract Interpretation. POPL 2000: 12-25

[110] Anca Browne, Edmund M. Clarke, Somesh Jha, David E. Long, Wilfredo R. Marrero: An Improved Algorithm for the Evaluation of Fixpoint Expressions. Theor. Comput. Sci. 178(1-2): 237-255 (1997)