Hybrid process algebra

Hybrid Process Algebra*

P.J.L. Cuijpers and M.A. Reniers

July 3, 2003

1 Introduction

1.1 Hybrid Systems

The theory of hybrid systems, studies the combination of continuous/physical and discrete/computational behaviour. When computational software is combined with mechanical and electrical components, or is interacting with, for example, chemical processes, a hybrid system arises in which the interaction between the continuous behaviour of the components, and the discrete behaviour of the software is important.

In current practice, often the discrete part of a hybrid system is described and analyzed using methods from computer science, while the continuous part is handled by control science. The design of the complete system is usually such that interaction between the discrete and continuous part is suppressed to a minimum. Because of this suppressed interaction, analysis is possible to some extent, but it limits the design options. In the field of hybrid systems theory, researchers attempt to extend the possibilities for interaction. The goal of this paper, is to develop an algebraic theory, called hybrid process algebra (HyPA), to support these attempts. Our hopes are that hybrid process algebra can serve as a mathematical basis for improvement of the design strategies of hybrid systems, and the possibility to analyse them. Further on in this introduction, we will review some of the already existing hybrid formalisms, and point out the defects of those, on which we hope to make some improvements.

In figure 1, a graphical representation is given of the general aim of our efforts. The figure shows our desire, that a hybrid theory is, in a sense, a conservative extension of computer science and systems theory. More precisely, a model from systems theory or computer science, should be expressible, and preferably look the same, in the hybrid theory, and theorems from systems theory and computer science should be transferable to the hybrid theory (when restricted to models from the original field of course). What the figure does not show, is that this conservativity is not the only goal. In that case, a simple

*We would like to express our gratitude to Progress/STW (Grant EES5173), Philips-CFT and Assembleon, for their financial and material support of this project.

union of the theories would be sufficient. We also desire a certain interaction between the theories, reflecting the interaction between software and physics described before. This goal is harder to formalize, but in the remainder of this introduction we hope to give some feeling for it, using examples of deficiencies (in our view) in existing hybrid formalisms, and indicating how we intend to improve on those.

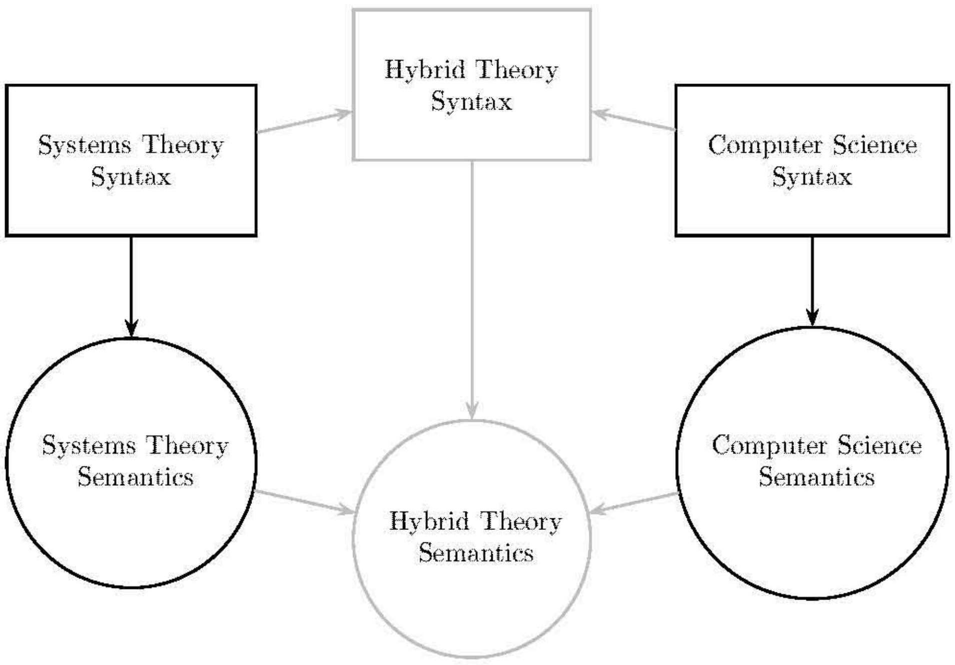

Figure 1: Developing Hybrid Theory

1.2 Algebraic Reasoning

In systems theory, algebraic reasoning is acknowledged by most people, as one of the most powerful tools available for analyzing physical behaviour. This behaviour is usually described by differential equations and inclusions, which model the rate of change of the value of certain continuous variables, or by difference equations and inclusions, modelling discrete changes in variables. Note, however, that these two ways of descriptions are hardly ever combined. As an example of a differential equation, $\dot{x} = f(x, u)$, in which x and u are variables ranging over the real numbers, and f is real valued function, models that the value of x changes continuously through time (indicated by the dot in $\dot{x}$) with a rate defined by $f(x, u)$, i.e. by the current value of x and u. Alternatively, if there is a choice of rates of change, one may write $\dot{x} \in F(x, u)$, in which F is a set-valued function over the reals. Also, an inequality $x \leq f(x, y)$ may denote that x is constrained in its value (not its rate of change) for some reason. As an example of a difference equation, $x^+ = f(x^-, u^-)$ denotes that the value of x is reassigned to $f(x^-, u^-)$, based on the previous values of x and u. This way of notation is for example used in [41]. Following Henzinger's way [26] of describing continuous behaviour, we allow any predicate over model variables $\mathbb{V}_m$ and their 'dotted' versions $\dot{\mathbb{V}}_m$ in this report. These predicates are called flow-clauses. Analogously, predicates over $\mathbb{V}_m^-$ and $\mathbb{V}_m^+$ are called 'reinitialization clauses', and are used to model discontinuous behaviour of model variables. However, although we allow arbitrary predicates over model variables, the analysis of systems will often turn out to be easier, if we confine ourselves to clauses based on some algebraic way of description.

In computer science, the usefulness of algebra is still a topic of much debate, but nevertheless there are interesting examples of applications of process algebra (see for example [23] for a list of references to protocol verifications, [12] for a start in the description and analysis of other industrial size problems, like the design of a controller for a coating system and a turntable system, and [22] for the description and analysis of railway interlocking specifications). In process algebra, the discrete actions that a system may perform are often considered atomic elements of the algebraic description language. These 'atomic actions' can be combined using operations describing choice between behaviors, sequential execution of behaviors, and concurrent execution of behaviors. For example, the process algebraic equation $X = aX + b$ describes a recursive process X that may choose to execute an atomic a action, followed by an execution of X again, or choose $(+)$ to execute b and terminate. In this report we take the process algebra ACP [8] and extend it with new atoms, describing continuous behaviour through the use of flow-clauses and discontinuous behaviour through reinitialization-clauses as mentioned before. Also, we import the disrupt operator from LOTOS [13], since it turned out to model the sequential composition of flow-clauses well. The choice for ACP is rather arbitrary, and we expect that the methods described in this report can be easily extended to other process algebras. Obviously, the choice for ACP leads to the need for an alternative notation for some symbols, since, amongst others, the process algebraic $+$ has a different meaning than the system theoretic $+$. In section 2.1 of this report, these notational problems are solved, and the notation for our hybrid process algebra is formalized.

Returning to figure 1, we may conclude that we have chosen the process algebra ACP as a representative syntax for computer science, and a predicate variant of differential equations and difference equations for systems theory. The resulting conservative extension, we call hybrid process algebra (or for short HyPA), in which abstract actions are described by the actions of standard process algebra, physical behaviour is described by flow clauses, and assignments and other discontinuities are described by reinitialization clauses. The reason for this choice of syntax, is that we agree that algebraic reasoning is an important tool for the analysis of systems, and that we would like to support the possibility of algebraic reasoning about hybrid systems. So far, the only algebraic approaches that we know of regarding hybrid systems, are described in [37, 39, 36] (hybrid χ), [44, 11] (hybrid versions of ACP), [27] (hybrid CSP) and [35] (ϕ-calculus). In the remainder of this introduction, we will explain the deficiencies that these methods have, in our opinion, in describing hybrid interaction. We should note, that within other hybrid formalisms like hybrid automata [26, 28], hybrid Petri-nets [9, 16, 19] and hybrid action systems [33], the use of algebraic reasoning on differential equations for analysis purposes, is not uncommon. It is the process algebraic reasoning that is underexposed. For a translation of hybrid automata into the process algebras CSP, and timed μCRL, see [2] and [45, 24], respectively.

In the hybrid theory that has been developed by system theorists (see for example [41, 42, 10, 25, 40, 18]) algebraic reasoning is possible, but none of these theories support reasoning about non-determinism. All of these theories have a trace semantics, and cannot distinguish deadlocks and non-deterministic choices. Since we would like a conservative extension of process algebra, we would also like to be able to distinguish systems up to the notion of bisimulation equivalence, and therefore, we consider the system theoretic formalisms as non conservative with respect to computer science.

In section 3 of this report, we prove formally that HyPA is a conservative extension of the process algebra ACP, and by construction of the semantics, it is immediately clear that it is a conservative extension of differential inclusions, at least up to the type of solutions we have chosen to use.

1.3 Hybrid Interaction

Many of the hybrid formalisms that we have mentioned so far, have some problem in the definition of parallel composition. And surprisingly, in most cases, this problem comes to light in a purely continuous case study. Let us consider the following example, depicted in figure 2, of a continuous plant P described by the differential equation $\dot{x} = f(x, u)$, and a continuous controller C described by $u = g(x)$. The composition of plant and controller will be denoted as $P \parallel C$.

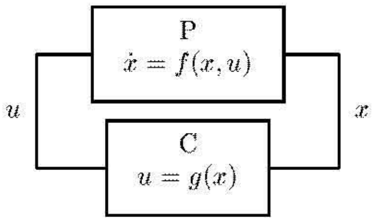

Figure 2: Continuous control system

The hybrid automata of Henzinger [26], as well as the hybrid process algebra of Vereijken [44], and of Jifeng [27], assume that the continuous behaviour of two composed systems is independent. Using these formalisms, the system $P \parallel C$ would not model any interaction between P and C at all, since the only interaction between systems can be through computational actions. The variable x of P would simply be regarded different from the variable x of C. Hence, in our opinion, these formalisms cannot be considered to be a conservative extension of systems theory. At least, they do not support the way in which we would like to think about parallel composition of systems.

In a similar way, it turns out that the parallel composition of the above processes is not defined for the hybrid automaton model of Lynch, Segala and Vaandrager [28]. At least, not without a few amendments. In the formalism of [28], it is necessary to identify variables as either state variables of a system, or as external variables of the system. These two sets of variables are supposed to be disjoint. The intuition behind this partition is that the state variables model the memory of the system, while the external variables model the communication with other systems. Therefore, in a parallel composition, it is required that two hybrid automata are *compatible*, meaning that the state variables of the one automaton do not intersect with any of the variables of the other automaton. Now, looking at the plant P of figure 2, we see that we need to choose x to be a state variable, otherwise information on x is lost between transitions, but it also needs to be an external variable, since we need to communicate its value with the controller C. This contradicts the requirement on hybrid automata that the set of state-variables and the set of external-variables are disjoint. Admittedly, the problem is not as big as it may seem, since by adding an external variable y, and the equation $y = x$, to the description of P, and changing the description of C to $u = g(y)$, we can declare x to be a state-variable, and find that the systems have become compatible. So, although the system in figure 2 cannot be modelled as $P \| C$ directly in this hybrid automaton model, we can model the slight modification depicted in figure 3 instead.

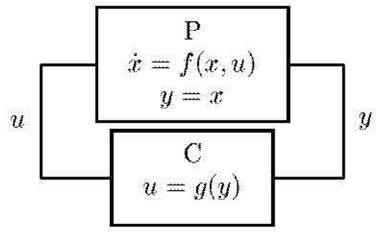

Figure 3: Compatible continuous control system

In [32] it was already noted that the variable-partition of a system, is not always uniquely determined by the equations that describe the system. Even in our simple control example, it is possible to use the equations $x = y$ and $u = g(x)$, and declare x external and y a state variable. Often, there is no clear physical ground to choose a specific partition. This is the one reason why we would like to avoid the partitioning of the set of variables of a system, in our semantics. Another reason, is that in basic textbooks on control theory (for example [17]), one usually starts out with developing controllers for plants of which the state-variables are also output-variables. It therefore seems, that the intuition behind compatibility, that state-variables do not play a role in communication with other systems, does not coincide with the system-theoretic intuition. This is confirmed by the theory discussed in [32], where state-variables may also be output-variables of a system, while external-variables may be inputs or outputs.

In this report, we show that partitioning the model variables as done for hybrid automata, is in fact not necessary, if a subtly different semantical view is taken. We have to credit the authors of [28] however, for being able to do analysis, to some extend, in the light of abstraction of variables (making them internal). The hybrid process algebra we present in this report is not concerned with any form of abstraction so far, because experience with normal process algebra shows that abstraction is a difficult topic to study algebraically, and we expect it to be convenient, that the basic theory is worked out first [7]. On the other hand, HyPA is developed in close cooperation with the people who are working on the formal semantics of the hybrid χ-language, which is focussed on the simulation of hybrid systems. Their operational semantics [36] uses a semantical structure similar to, and based on, the one we have developed for HyPA (discussed in section 2.2), and their language does contain and operator that allows for the hiding of model variables (although there is no axiomatization for it yet). Also the hybrid process algebra of Bergstra and Middelburg [11], that appeared as a technical report just one week before this one went to print, uses a hybrid transition system semantics, and has comparable definitions for parallel composition. Furthermore it has a form of abstraction from model variables, comparable to that of χ. We expect, therefore, that it is possible to develop a similar abstraction operator for HyPA, and hopefully to find a way to reason about it algebraically. In section 4, we discuss the relation between HyPA, χ and the process algebra of [11] in more detail. Admittedly, these three languages are very similar, which calls for a more thorough comparison in the near future.

In ϕ-calculus [35], the semantics assumes continuous behaviour to be a property of the environment, rather than of the process itself. There, (urgent) environmental actions allow the process to change the rules for continuous behaviour in an interleaving manner, which leads to the replacement of one differential equation by another. Again, there is no continuous interaction between P and C. When we write $P \,||\, C$ in ϕ-calculus, the semantics is such that only the continuous behaviour of the plant or of the controller is executed. This, clearly, contradicts with our intuition on the parallel composition.

In hybrid action systems, the parallel composition of P and C leads to the desired result, if we ignore some irrelevant syntactical constructs. However, the parallel composition of two differential equations $\dot{x} = 1 \,||\, \dot{x} = 2$ results in a process that acts like the differential inclusion $\dot{x} \in \{1, 2\}$. This, again, contradicts with our intuition. We would expect contradicting equations to result in deadlock. Nevertheless, both the 'interleaving' approaches from ϕ-calculus and hybrid action systems, might turn out to be useful in situations where our intuition is flawed, and the theories might be considered complementary to HyPA.

In conclusion, we might state that we aim for an algebraic formalism, in which the parallel composition has a similar intuition as in [28], but without having to require compatibility of the composed systems. To do this, we have worked out the notion of ***hybrid transition system***, as a semantical framework, in [15]. This framework unifies the discrete behaviour of computer science and the continuous behaviour of system theory in a similar way as the hybrid automata of [28] do, while avoiding the explicit use of state variables and external variables. From a system theoretic point of view, hybrid transition systems are an extension of Sontag machines [38]. Returning to figure 1, one might say that the chosen semantics of the original fields are transition systems for computer science, and Sontag machines for system theory. Our conservative extension of those is called hybrid transition system. On the framework of hybrid transition systems, it turns out to be rather easy to define an operational semantics for actions, as well as for differential equations and inclusions. Also all kinds of compositions known from process algebra, among which parallel composition, can be defined easily using the method for giving an operational semantics introduced in [31]. As far as we know, HyPA and χ and the process algebra of [11] are the only process algebras for hybrid systems so far, that use an operational semantics in which complete physical signals are taken into account rather than only the time-behaviour of a system.

1.4 Discontinuities

Regarding discontinuous behaviors, the semantics for differential equations in HyPA, differs a little from the usual interpretation taken in, for example, Henzinger's hybrid automata. The standard approach there (and in most other hybrid formalisms), is to assume only continuous behaviour of all variables, unless they are specifically altered by assignment transitions. For some hybrid descriptions of physical behaviour, however, it is convenient that certain variables can also behave discontinuously. Take, for example, the electrical circuit depicted in figure 4, in which a switch steers the voltage over a resistor-capacity combination.

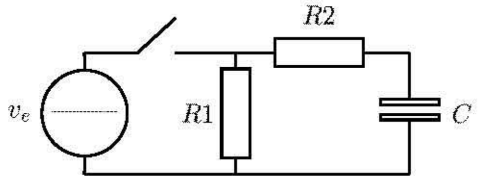

Figure 4: Electrical circuit with switch

For such a system, it is desirable to model the voltage over, and the current through the resistors (v_{R1}, v_{R2}, i_{R_1} and i_{R2}) as discontinuous functions of time. A possible hybrid automaton model for this circuit, is depicted in figure 5. Note, that there are arbitrary jumps modelled on the transitions, for the discontinuous variables (i.e. not for v_C!). This is necessary, because, without deeper analysis of the differential equations, we do not know what kind of discontinuities may occur. In order to avoid discontinuous behaviour that violates the physical properties of the circuit, we may indicate in the hybrid automaton model, that the algebraic equations used to describe the electrical circuit are *invariants*. As an example of an undesired discontinuity, one should note that, when the switch closes, the current through the second resistor (i_{R2}) is determined completely by the source voltage v_e and the voltage over the capacitor v_C. The invariants make sure that no other assignments can be made to i_{R2}.

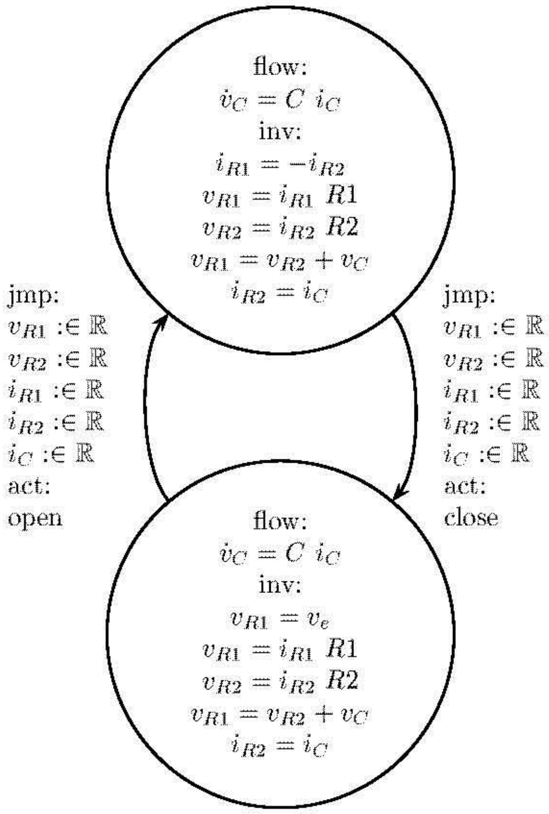

Figure 5: Automaton modelling the electrical circuit

Now, in the case of higher index differential equations, the approach of using invariants to avoid undesired discontinuities breaks down. As an example, let us consider a system described by the following equations, in which z is a variable that may behaviour discontinuously:

$$\begin{aligned} \dot{x} &= z \\ \dot{y} &= -z \\ x &= y\ . \end{aligned}$$

As before, it is undesirable that an assignment to z is made that violates these equations. But the approach that is usually taken in hybrid automata theory, to take all algebraic equations to be invariants, does not work here. The choice of z is independent from the choice of x and y. Clearly, the system only can perform continuous behaviour, if the value of z is reset immediately to zero. This, however, is insight obtained through analysis of the equations, and should therefore not be used when modelling a system. As far as we know, there is no solution in hybrid automaton theory for this problem. This is why we take a different approach regarding discontinuous behaviour in HyPA.

In HyPA, when modelling a differential equation, we can indicate explicitly whether z is continuous or discontinuous. If it is continuous, we find deadlock for the higher index differential equations of the previous example. If it is discontinuous, only those discontinuities can occur for which a solution exists. Furthermore, assignments are modelled, not as a kind of atomic actions (as with hybrid automata), but as reinitializations of processes. As it turns out, the reinitialization of flow-clauses only occurs if the flow-clause has a solution after reinitializing, while the reinitialization of atomic actions is always executable. This will be explained in more detail in section 2.2 of this report.

1.5 Drawbacks

At first sight, there seem to be two major drawbacks to our method. The first drawback, is that we need a kind of bisimulation equivalence that takes into account the valuation of all variables, in order for it to be a congruence for parallel composition. However, this does not render the whole theory useless, because the same method of requiring compatibility of processes that was used in [28] in order to define parallel composition, can be used in HyPA to guarantee congruence of parallel composition under a weaker notion of equivalence (like the one used in [28]), and furthermore, we give an axiomatization for our notion of equivalence that allows elimination of the parallel composition from closed process terms, so that weaker notions of equivalence can be used for analysis of processes after applying this elimination. The second drawback, is that some of the axioms become rather confusing due to the discontinuities that may be possible in some of the variables of a differential equation. This can be helped, as we show in section 3, by simply requiring all variables to be continuous, as in hybrid automata. So, in conclusion, the theory is not more difficult or cumbersome, if we model processes under the usual restrictions. In fact, as we indicate in section 4.1, we expect that HyPA is a conservative extension of hybrid automata, although we do not give a formal proof of this claim. Furthermore, we have new constructs to our disposition that are not available, yet, in other hybrid formalisms, at the cost of having to use more difficult axioms.

1.6 Structure of this report

In section 2.1 of this report, the syntax of HyPA is presented, describing how the standard process algebra ACP [8] is extended with a constant for termination, the so-called disrupt operator, known from LOTOS [13], and a simplification of the two types of clauses from [41], representing continuous and discontinuous behaviour. As we mentioned before, HyPA does not contain any operators for handling abstraction of actions or variables yet. But, even without abstraction constructs, hybrid process algebra turns out to be interesting topic of study, already. Furthermore, since abstraction is a rather complicated subject, it seems wise to develop a concrete process algebra first [7]. In section 2.2, a hybrid transition system semantics is defined in the style of [31], in which continuous behaviour is synchronizing, and discrete behaviour is interleaving. Section 3 is devoted to an axiomatization of HyPA, for the equivalence notion of bisimulation [29]. In this section, also the formal relation with ACP is discussed, and a set of basic terms is given into which closed HyPA terms can be rewritten. We conclude by giving our own views on the work presented, and by making suggestions for future research. In the appendices, we give the proofs for the soundness, conservativity and rewriting claims of section 3.

2 Hybrid Process Algebra

2.1 Syntax

In this section, the syntax of HyPA is introduced, which is an extension of the process algebra ACP [8, 21], with the disrupt operator from LOTOS [13] and with flow-clauses and reinitialization-clauses from the event-flow formalism introduced in [41]. The syntax of HyPA is defined by:

$$\begin{array}{rcl} P & ::= & \delta \mid \epsilon \mid a \mid c \mid d \gg P \mid P \oplus P \mid P \odot P \mid \\ & & P \blacktriangleright P \mid P \rhd P \mid P \parallel P \mid P \mathbin{\|\!\lfloor} P \mid P \,|\, P \mid \partial_H(P) \ , \end{array}$$

where $a \in A$, $c \in C$ and $d \in D$ are not defined formally here, but only described informally in the following explanatory text.

- There are the atomic process terms δ (called *deadlock*) and ϵ (called *empty process*), modelling respectively a deadlocking process and a (successfully) terminating process.
- There are atomic *discrete actions*, chosen from a set A, used to model discrete, computational behaviour.
- There are *flow-clauses*, used to model continuous, never terminating, physical behaviour. Flow-clauses are chosen from a set C, formed by pairs

$[\,V \mid \text{Pred}\,]$ of predicates Pred, in which model variables $\mathbb{V}_m$ (with $x \in \mathbb{V}_m$ taking value in $\mathcal{V}(x)$) and their derived[1] versions $\dot{\mathbb{V}}_m = \{\dot{x} \mid x \in \mathbb{V}_m\}$ (with $\dot{x}$ also taking value in $\mathcal{V}(x)$) may occur as free variables, and of a subset $V \subseteq \mathbb{V}_m$, signifying which variables are continuous, in the sense that during the evolution of time their value may not jump. We usually leave out the brackets for V, and even omit it (and the '$|$' delimiter) if it is empty. Typical flow-clauses are differential equations, for example $[\,x \mid \dot{x} = f(x, y)\,]$, and algebraic inequalities, for example $[\,x \leq f(x, y)\,]$. Furthermore, the set C is closed under conjunction ($\wedge$) of clauses. In general, the domain $\mathcal{V}(x)$ of a model variable $x \in \mathbb{V}_m$ is specified by the modeller at the first introduction of the variables. In this paper, the specification of domains is usually left out since, most of the time, it is obvious from the context. We write $\mathcal{V} = \bigcup_{x \in \mathbb{V}_m} \mathcal{V}(x)$ for the union of all variable domains, and $Val = \mathbb{V}_m \rightarrow \mathcal{V}$ for the set of variable valuations.

- There are *reinitialization-clauses*, used to model discontinuous changes in the values of the model variables. Reinitialization-clauses are chosen from a set D, formed by pairs $[\,V \mid \text{Pred}\,]$, consisting of predicates Pred, in which free variables may occur from the sets $\mathbb{V}_m^- = \{x^- \mid x \in \mathbb{V}_m\}$ and $\mathbb{V}_m^+ = \{x^+ \mid x \in \mathbb{V}_m\}$, modelling the current and future value of a model variable, respectively, and of a set $V \subseteq \mathbb{V}_m$ modelling which variables are allowed to change. Typical reinitialization-clauses are assignments, for example $[\,x \mid x^+ = f(x^-, y^-)\,]$ which, in imperative programming, is usually denoted as $x := f(x, y)$. But, also boolean predicates can be modelled, by choosing V empty (using the same notational conventions as with flow-clauses), and only using the current value of variables, i.e. $[\,\text{Pred}(x^-, y^-, \dots)\,]$. Apart from containing the above described predicates, the set D is closed under conjunction ($\wedge$), disjunction ($\vee$), and concatenation ($\sim$) of reinitialization-clauses. Also, there is a satisfiability operator ($d^?$) on clauses $d \in D$, which does not reinitialize the values of a model variable, but only executes the reinitialized process, if d can be satisfied in some way, and there is a reinitialization-clause (c_{jmp}) derived from a flow-clause $c \in C$, which executes the same discontinuities that are allowed initially by the flow-clause. These last two operators turn out to be especially useful when calculating with process terms. Reinitializations-clauses are assumed to act upon another process, so if $p \in P$ is a HyPA process, and $d \in D$ is a reinitialization-clause, then $d \gg p$ denotes the reinitialized HyPA process.

- There is the *alternative composition*[2] $p \oplus q$, modelling a (non-deterministic)

[1]We assume derivation is defined for all variables, but if we want to us a variable x for which this is not the case (for example a computational data structure), then no formal problems arise as long as we do not use the derived variable $\dot{x}$ in our predicates.

[2]We use the notation $\oplus$ and $\odot$ for alternative and sequential composition, rather than the usual $+$ and $\cdot$, to avoid confusion with the notation used in flow-clauses and reinitialization-clauses for addition and multiplication. Also, we use $X \approx p$ for recursion rather than $X = p$. We realize that this might distract people in the field of process algebra, yet chose to adapt

choice between the processes p and q.

- There is the *sequential composition* $p \odot q$, modelling a sequential execution of processes p and q. The process q is executed after termination of the process p.

- There is the *disrupt* operator $p \blacktriangleright q$, which models a kind of sequential composition where the process q may take over execution from process p at any moment, without waiting for its termination. This composition is invaluable when modelling two flow-clauses executing one after the other, since the behaviour of flow-clauses is ongoing, and never terminates. The disrupt is originally introduced in the language LOTOS [13], where it is used to model for example exception-handling. Also, it is used, for example in [4], for the description of mode-switches.

- Related to the disrupt, there is the *left-disrupt* operator, denoted $p \rhd q$, which first executes a part of the process p and then behaves as a normal disrupt. The left-disrupt is mainly needed for calculation and axiomatization purposes, rather than for modelling purposes. For example, it occurs often when we attempt to eliminate the parallel composition from a process term through axiomatic reasoning, as described in section 3.

- There is the *parallel* composition, $p \parallel q$, which models concurrent execution of p and q. The intuition behind this concurrent execution is that discrete actions are executed in an interleaving manner, with the possibility of synchronization (as in ACP, where synchronization is called communication), while flow-clauses are forced to synchronize, and can only synchronize if they accept the same solutions. The synchronization of actions takes place using a (commutative and associative) communication function $\gamma \in (A \times A) \to A$. For example, if the actions a and a' synchronize, the resulting action is $a'' = a \gamma a'$. Actions cannot synchronize with flow-clauses, and in a parallel composition between those, the action will execute first.

- Related to the parallel composition there are the operators $p \mathbin{|\!\lfloor} q$ (called *left-parallel composition*), which denotes that p performs a discrete action first (if possible), and then it behaves as normal parallel composition, and $p \mid q$ (called *forced-synchronization*), that denotes how the first behaviour (either a discrete action or a part of a flow) of p and q is synchronized, after which it behaves as normal parallel composition. As with the left-disrupt, these operators are mainly introduced for calculation purposes.

- There is the *encapsulation*, $\partial_H(p)$, which models that certain discrete actions (from the set $H \subseteq A$) are blocked during the execution of the process p. This operator is often used in combination with the parallel

the process algebraic notation rather than the notation adopted from system theory, simply because the latter has been in use for a longer time already.

composition to model that synchronization between discrete actions is enforced.

- Finally, all the processes should be interpreted in the light of a set E of recursive definitions of the form $X \approx p$, where X is a recursion variable from the set $\mathbb{V}_r$ (with $\mathbb{V}_r \cap \mathbb{V}_m = \emptyset$) and p is a process term in which this, and other recursion variables, may occur. Recursion is a powerful way to model repetition in a process.

The binding order of the operators of HyPA is the following: $\odot$, $\blacktriangleright$, $\rhd$, $d \gg$, $\|$, $\lfloor\!\lfloor$, $|$, $\oplus$, where alternative composition binds weakest, and sequential composition binds strongest. With encapsulation (∂_H (_)), brackets are always used. As an example, a term $d \gg a \odot b \oplus c \,\|\, c'$ should be read as $(d \gg (a \odot b)) \oplus (c \,\|\, c')$.

2.2 Formal Semantics

In this section, we give a formal semantics to the syntax defined in the previous section, by constructing a kind of labelled transition system, for each process term and each possible valuation of the model variables. In this transition system we consider two different kinds of transitions: one associated with computational behaviour (i.e. discrete actions), and the other associated with physical behaviour (i.e. flow-clauses). This is why we call those transition systems hybrid.

Definition 1 (Hybrid Transition System) *A* hybrid transition system *is a tuple* $\langle X, \mathbb{A}, \Sigma, \mapsto, \leadsto, \checkmark \rangle$, *consisting of a* state space X, *a set of actions labels* $\mathbb{A}$, *a set of signal labels* Σ, *and transition relations* $\mapsto \subseteq X \times \mathbb{A} \times X$ *and* $\leadsto \subseteq X \times \Sigma \times X$. *Lastly, there is a termination predicate* $\checkmark \subseteq X$.

For the semantical hybrid transition systems that are associated with HyPA terms, the state space is formed by pairs of process terms and valuations of the model variables to their domain, i.e. $X = P \times \mathit{Val}$. The set of action labels is formed by pairs of actions and valuations, i.e. $\mathbb{A} = A \times \mathit{Val}$, and the set of signal labels is formed by partial functions of time to valuations, i.e. $\Sigma = T \mapsto \mathit{Val}$. In this paper, we model time using the non-negative real numbers ($T = \mathbb{R}^{\geq 0}$). Furthermore, we limit ourselves to those elements $\sigma \in \Sigma$ that have a closed interval, non-singleton domain, which includes the least element 0, so if $\sigma \in \Sigma$ then the domain of σ is of the form $\mathit{dom}(\sigma) = [0 \ldots t]$, for some $t > 0$. We use the notation $\langle p, \nu \rangle \overset{a,\nu'}{\mapsto} \langle p', \nu'' \rangle$ for a transition $((p,\nu),(a,\nu'),(p',\nu'')) \in \mapsto$ with $(p,\nu),(p',\nu'') \in X$ and $(a,\nu') \in \mathbb{A}$. Similarly, we use $\langle p, \nu \rangle \overset{\sigma}{\leadsto} \langle p', \nu' \rangle$ for a transition $((p,\nu),\sigma,(p',\nu')) \in \leadsto$ with $\sigma \in \Sigma$, and for arbitrary transitions, we use $\langle p, \nu \rangle \xrightarrow{l} \langle p', \nu' \rangle$ instead of $((p,\nu),l,(p',\nu')) \in \mapsto \cup \leadsto$ and $l \in \mathbb{A} \cup \Sigma$. Finally, termination is denoted $\langle p, \nu \rangle \checkmark$ instead of $(p,\nu) \in \checkmark$.

Hybrid transition systems can be used to model computational behaviour through the use of action transitions, which take no time to execute, and to model physical behaviour through the use of signal transitions, which represent the behaviour of model variables during the passage of time. Note, that there is no variable in $\mathbb{V}_m$ that is explicitly associated with time. Hence, if one would like to refer to time in a flow-clause, one would have to include the model of a clock, using for example a differential equation like $\left[\, t \mid \dot{t} = 1 \,\right]$.

In table 1, the semantics of the atomic processes and the flow-clauses is given, using deduction rules in the style of [31]. In rule number (3),(4) and (5) we use the notations $(\nu, \sigma) \models c$ and $(\nu, \nu') \models d$, for notions of solution for flow-clauses and reinitialization-clauses, that are explained further on in this section.

$$\frac{}{\langle\, \epsilon, \nu \,\rangle \checkmark}(1) \qquad \frac{}{\langle\, a, \nu \,\rangle \overset{a,\nu}{\mapsto} \langle\, \epsilon, \nu \,\rangle}(2) \qquad \frac{(\nu, \sigma) \models c,\ \ dom(\sigma) = [0 \ldots t]}{\langle\, c, \nu \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, c, \sigma(t) \,\rangle}(3)$$

$$\frac{(\nu, \nu') \models d,\ \langle\, p, \nu' \,\rangle \checkmark}{\langle\, d \gg p, \nu \,\rangle \checkmark}(4) \qquad \frac{(\nu, \nu') \models d,\ \langle\, p, \nu' \,\rangle \xrightarrow{l} \langle\, p', \nu'' \,\rangle}{\langle\, d \gg p, \nu \,\rangle \xrightarrow{l} \langle\, p', \nu'' \,\rangle}(5)$$

Table 1: Operational semantics of HyPA

Rule (1) captures our intuition that ϵ is a process that only terminates. Analogously, the fact that there is no rule for δ, expresses that this is indeed a deadlocking process. Rule (2) expresses that discrete actions display their own name, and the valuation of the model-variables on the transition label, but do not change this valuation. Changes in the valuation can only be caused by flow-clauses and reinstallation-clauses, as defined by rules (3) to (5).

A flow-clause changes the valuation of the model-variables according to the possible solutions of its predicate. In contrast to the flow predicates of [26], discontinuities in the flow of a variable x, are allowed in HyPA when $x \notin V$. Formally, we define the concept of solution of a flow-clause as follows.

Definition 2 (Derived signal) *For a signal $\sigma \in \Sigma$, the derived signal $\dot{\sigma}$, on the same domain, is defined by $\dot{\sigma}(t')(x) = \left(\frac{d}{dt}\sigma(t)(x)\right)(t')$, for all $t' \in dom(\sigma)$. On the bounds of the domain, we use the left- and right-derivative, respectively.*

Definition 3 (Solution of a flow-predicate) *We define that a pair $(\nu, \nu') \in Val \times Val$ is a solution of a predicate Pred on free variables $\mathbb{V}_m \cup \dot{\mathbb{V}}_m$, denoted $(\nu, \nu') \models Pred$, if the predicate evaluates to true, when every variable x in the predicate is evaluated to $\nu(x)$, and every derived variable $\dot{x}$ in the predicate is evaluated to $\nu'(x)$.*

Definition 4 (Solution of a flow-clause) *A pair $(\nu, \sigma) \in Val \times \Sigma$, is defined to be a* solution *of a flow-clause $c \in C$, denoted $(\nu, \sigma) \models c$, as follows.*

- *$(\nu, \sigma) \models \left[\, V \mid Pred \,\right]$ if for all $t \in dom(\sigma)$ we find $(\sigma(t), \dot{\sigma}(t)) \models Pred$, and for all $x \in V$ we find $\nu(x) = \sigma(0)(x)$ and $\sigma(\cdot)(x)$ is a continuous function from T to $\mathcal{V}(x)$;*

- $(\nu, \sigma) \models c \wedge c'$ *if* $(\nu, \sigma) \models c$ *and* $(\nu, \sigma) \models c'$;

Clearly, the clause $[\,\textit{false}\,]$ has no solutions. Furthermore, for the solutions of a differential equation $[\,x \mid \dot{x} = f(x,y)\,]$, we find that $\sigma(\cdot)(x)$ (with x the differentiated variable) is a continuous and differentiable function of time. For $[\,\textit{true}\,]$, every signal (including discontinuous signals) is a solution.

Reinitialization-clauses change the valuation of the model-variables according to the possible solutions of their predicate. The set V, that we use in addition to predicates, indicates that a variable is allowed to change its value. Whenever $x \notin V$, the variable x is fixed, and we can extend the predicate with $x^- = x^+$. Formally, we define a solution of a reinitialization-clause as follows.

Definition 5 (Solution of a reinitialization-predicate) *We define that a pair* $(\nu, \nu') \in \mathit{Val} \times \mathit{Val}$ *is a solution of a predicate Pred on free variables* $\mathbb{V}_m^- \cup \mathbb{V}_m^+$, *denoted* $(\nu, \nu') \models \mathit{Pred}$, *if the predicate evaluates to true, when every variable* x^- *in the predicate is evaluated to* $\nu(x)$, *and every derived variable* x^+ *in the predicate is evaluated to* $\nu'(x)$.

Definition 6 (Solution of a reinitialization-clause) *A pair* $(\nu, \nu') \in \mathit{Val} \times \mathit{Val}$ *is defined to be a* solution *of a reinitialization-clause* $d \in D$, *denoted* $(\nu, \nu') \models d$, *as follows.*

- $(\nu, \nu') \models [\,V \mid \mathit{Pred}\,]$ *if* $(\nu, \nu') \models \mathit{Pred}$ *and for all* $x \notin V$ *we find* $\nu(x) = \nu'(x)$.
- $(\nu, \nu') \models d' \vee d''$ *if* $(\nu, \nu') \models d'$ *or* $(\nu, \nu') \models d''$;
- $(\nu, \nu') \models d' \wedge d''$ *if* $(\nu, \nu') \models d'$ *and* $(\nu, \nu') \models d''$;
- $(\nu, \nu') \models d' \sim d''$ *if there exists* $\upsilon \in \mathit{Val}$ *with* $(\nu, \upsilon) \models d'$ *and* $(\upsilon, \nu') \models d''$;
- $(\nu, \nu') \models d'^{?}$ *if* $\nu = \nu'$, *and there exists* $\upsilon \in \mathit{Val}$ *with* $(\nu, \upsilon) \models d'$.
- $(\nu, \nu') \models c_{jmp}$ *if there exists* $\sigma \in \Sigma$ *such that* $(\nu, \sigma) \models c$ *and* $\sigma(0) = \nu'$.

Obviously, $[\,\textit{false}\,]$ has no solutions, while $[\,\mathbb{V}_m \mid \textit{true}\,]$ has every possible reinitialization as solution. Note, that $[\,\textit{true}\,]$ exactly allows all reinitializations that do not change any of the variable valuations. If we want to model a conditional execution of the form "if Pred then p", for a predicate Pred on free variables from $\mathbb{V}_m$, and a process $p \in P$, this can be done keeping V empty and by writing $[\,\mathrm{Pred}^-\,] \gg p$. Here, Pred^- denotes that we replace every free variable x in Pred by x^-. In a similar way, we sometimes write Pred^+ to denote the replacement of every x with x^+. Taking V empty implies that, for all variables $x \in \mathbb{V}_m$, we have $x^- = x^+$. Finally, if we have a two reinitialization-clauses $d, d' \in D$, the clause $d \sim d'$ accepts exactly those solutions that are in some way a concatenation of the reinitializations of d and d'. The clause $d^{?}$ does not change the value of any of the variables, and only has a solution for those valuations for which d has a solution. The clause c_{jmp} imitates the reinitializations performed initially by a flow-clause c.

Now that the atomic behaviour of HyPA has been explained, let us take a closer look at the operators. Their semantics is defined in table 2. Rules (6) to (10), for alternative and sequential composition, are very similar to that of ACP. However, it is worth noting that we have chosen to model signal-transitions as having the same non-deterministic interpretation as action-transitions. This in contrast to many timed process algebras [5], where the passage of time (by itself) does not trigger a branching in the transition system. The reason for this way of modelling, is our intuition that continuous behaviour (i.e. the passing of time) influences the valuation of the model variables, and can therefore introduce choices in the system behaviour, just like discrete actions do. If, in the future, we develop operators to abstract away from the variables that trigger those choices, we do not want the choices themselves to disappear, through some time-determinism mechanism. The argument for introducing time-determinism, that time is an external phenomenon, does in our opinion not hold for hybrid systems. Also, the hybrid automata of Henzinger [26], and most other hybrid automata approaches that we know of, are time-non-deterministic, supposedly for the same reasons.

Interestingly, in [11] a time-deterministic approach to hybrid systems is chosen (clearly they disagree with the above arguments), while in χ [36] an operator is introduced for both. Models in the χ languages, therefore, might show the difference between the approaches. And, as far as we can tell, the time-deterministic operator is used most often when, for example, a controller makes a choice after some delay. This is modelled as a time-deterministic choice between delaying actions. When modelling physical modes of a system, the non-deterministic choice operator is used. The physical behaviour of a system can only be in one mode, even if a particular evolution is permitted in both modes.

Rules (11) to (14) define the semantics of the disrupt operator, and the left-disrupt operator. If we compare these rules to the rules for sequential composition, we see that the main difference, is the way in which termination is handled. Firstly, in a composition $p \blacktriangleright q$, the process q may start execution without p terminating. And secondly, if the process p terminates, the process $p \blacktriangleright q$ may also terminate regardless of the behaviour of q.

Rules (15) to (19) define the semantics of the parallel composition, and in these rules the difference between action and signal transitions is most prominent. For actions, the interpretation of the parallel composition is the same as in ACP [8, 21]. Discrete actions that are placed in parallel are interleaved, but can also synchronize using a (partial, commutative and associative) communication function $\gamma \in A \times A \mapsto A$. If a discrete action a communicates with an action a', the valuation of the model variables has to be the same for both, and the result is an action $a'' = a\gamma a'$. If flow-clauses are placed in parallel, they always synchronize their behaviour such that, intuitively, the signals that are possible in a parallel composition are a solution of both clauses.

The encapsulation, as defined by rules (20) to (22), only influences action transitions. This is not surprising, since the $\partial_H(\_)$ operator is originally intended to model enforced synchronization in a parallel composition. Signal

$$\frac{\langle p,\nu\rangle\checkmark}{\begin{array}{c}\langle p\oplus q,\nu\rangle\checkmark\\ \langle q\oplus p,\nu\rangle\checkmark\end{array}}(6)\quad \frac{\langle p,\nu\rangle\xrightarrow{l}\langle p',\nu'\rangle}{\begin{array}{c}\langle p\oplus q,\nu\rangle\xrightarrow{l}\langle p',\nu'\rangle\\ \langle q\oplus p,\nu\rangle\xrightarrow{l}\langle p',\nu'\rangle\end{array}}(7)\quad \frac{\langle p,\nu\rangle\checkmark,\ \langle q,\nu\rangle\checkmark}{\langle p\odot q,\nu\rangle\checkmark}(8)$$

$$\frac{\langle p,\nu\rangle\xrightarrow{l}\langle p',\nu'\rangle}{\langle p\odot q,\nu\rangle\xrightarrow{l}\langle p'\odot q,\nu'\rangle}(9)\quad \frac{\langle p,\nu\rangle\checkmark,\ \langle q,\nu\rangle\xrightarrow{l}\langle q',\nu'\rangle}{\langle p\odot q,\nu\rangle\xrightarrow{l}\langle q',\nu'\rangle}(10)$$

$$\frac{\langle p,\nu\rangle\checkmark}{\begin{array}{c}\langle p\blacktriangleright q,\nu\rangle\checkmark\\ \langle p\rhd q,\nu\rangle\checkmark\end{array}}(11)\quad \frac{\langle p,\nu\rangle\xrightarrow{l}\langle p',\nu'\rangle}{\begin{array}{c}\langle p\blacktriangleright q,\nu\rangle\xrightarrow{l}\langle p'\blacktriangleright q,\nu'\rangle\\ \langle p\rhd q,\nu\rangle\xrightarrow{l}\langle p'\blacktriangleright q,\nu'\rangle\end{array}}(12)$$

$$\frac{\langle q,\nu\rangle\checkmark}{\langle p\blacktriangleright q,\nu\rangle\checkmark}(13)\quad \frac{\langle q,\nu\rangle\xrightarrow{l}\langle q',\nu'\rangle}{\langle p\blacktriangleright q,\nu\rangle\xrightarrow{l}\langle q',\nu'\rangle}(14)$$

$$\frac{\langle p,\nu\rangle\checkmark,\langle q,\nu\rangle\checkmark}{\begin{array}{c}\langle p\,\|\,q,\nu\rangle\checkmark\\ \langle p\,|\,q,\nu\rangle\checkmark\end{array}}(15)\quad \frac{\langle p,\nu\rangle\overset{\sigma}{\rightsquigarrow}\langle p',\nu'\rangle,\ \langle q,\nu\rangle\overset{\sigma}{\rightsquigarrow}\langle q',\nu'\rangle}{\begin{array}{c}\langle p\,\|\,q,\nu\rangle\overset{\sigma}{\rightsquigarrow}\langle p'\,\|\,q',\nu'\rangle\\ \langle p\,|\,q,\nu\rangle\overset{\sigma}{\rightsquigarrow}\langle p'\,\|\,q',\nu'\rangle\end{array}}(16)$$

$$\frac{\langle p,\nu\rangle\overset{\sigma}{\rightsquigarrow}\langle p',\nu'\rangle,\ \langle q,\nu\rangle\checkmark}{\begin{array}{c}\langle p\,\|\,q,\nu\rangle\overset{\sigma}{\rightsquigarrow}\langle p',\nu'\rangle\\ \langle q\,\|\,p,\nu\rangle\overset{\sigma}{\rightsquigarrow}\langle p',\nu'\rangle\\ \langle p\,|\,q,\nu\rangle\overset{\sigma}{\rightsquigarrow}\langle p',\nu'\rangle\\ \langle q\,|\,p,\nu\rangle\overset{\sigma}{\rightsquigarrow}\langle p',\nu'\rangle\end{array}}(17)\quad \frac{\langle p,\nu\rangle\overset{a,\nu'}{\mapsto}\langle p',\nu''\rangle}{\begin{array}{c}\langle p\,\|\,q,\nu\rangle\overset{a,\nu'}{\mapsto}\langle p'\,\|\,q,\nu''\rangle\\ \langle q\,\|\,p,\nu\rangle\overset{a,\nu'}{\mapsto}\langle q\,\|\,p',\nu''\rangle\\ \langle p\,\lfloor\!\lfloor\,q,\nu\rangle\overset{a,\nu'}{\mapsto}\langle p'\,\|\,q,\nu''\rangle\end{array}}(18)$$

$$\frac{\langle p,\nu\rangle\overset{a,\nu'}{\mapsto}\langle p',\nu''\rangle,\ \langle q,\nu\rangle\overset{a',\nu'}{\mapsto}\langle q',\nu''\rangle,\ a''=a\,\gamma\,a'}{\begin{array}{c}\langle p\,\|\,q,\nu\rangle\overset{a'',\nu'}{\mapsto}\langle p'\,\|\,q',\nu''\rangle\\ \langle p\,|\,q,\nu\rangle\overset{a'',\nu'}{\mapsto}\langle p'\,\|\,q',\nu''\rangle\end{array}}(19)$$

$$\frac{\langle p,\nu\rangle\overset{a,\nu'}{\mapsto}\langle p',\nu''\rangle,\ a\notin H}{\langle\partial_H(p),\nu\rangle\overset{a,\nu'}{\mapsto}\langle\partial_H(p'),\nu''\rangle}(20)$$

$$\frac{\langle p,\nu\rangle\overset{\sigma}{\rightsquigarrow}\langle p',\nu'\rangle}{\langle\partial_H(p),\nu\rangle\overset{\sigma}{\rightsquigarrow}\langle\partial_H(p'),\nu'\rangle}(21)\quad \frac{\langle p,\nu\rangle\checkmark}{\langle\partial_H(p),\nu\rangle\checkmark}(22)$$

$$\frac{\langle p,\nu\rangle\checkmark}{\langle X,\nu\rangle\checkmark}(23)\ X\approx p\in E\qquad \frac{\langle p,\nu\rangle\xrightarrow{l}\langle p',\nu'\rangle}{\langle X,\nu\rangle\xrightarrow{l}\langle p',\nu'\rangle}(24)\ \ X\approx p\in E$$

Table 2: Operational semantics of HyPA, continued

transitions are already synchronized.

Rules (23) and (24) model recursion in the same way as it was done in [8, 21]. For a recursive definition $X \approx p$, a transition for the variable X is possible, if it can be deduced from the semantical rules for the process term p.

2.3 Example: Steam Boiler

This section is intended to illustrate the use of HyPA for modelling hybrid systems. The process below, is a model of the celebrated benchmark problem of the steam boiler [1]. For reasons of brevity, the problem is simplified considerably. It is not our intention to give a comparison with other models of the steam boiler here. We only want to give a feeling for the syntax and semantics of the language. The text below, explains shortly what the given model consists of.

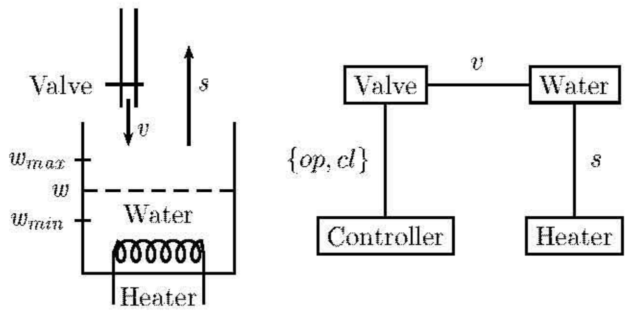

Figure 6: The steam boiler

The boiler process, as depicted in figure 6 consists of a water level w, an in-flow v and a steam production s. This stream production is determined by the Heater process, which limits it between s_{min} and s_{max}. The in-flow is determined by a Valve process, which can be opened or closed using the signals ro and rc respectively. If the valve is open, the in-flow to the boiler is v_{in}. If it is closed, the in-flow is 0. Furthermore, there is a Controller, that every T seconds interferes with the valve, by telling it to open or close using the signals so and sc. The goal of this controller, is to keep the water level between w_{min} and w_{max}. To do this safely, it takes a margin of w_{safe} into account. The total system is the parallel composition of the Water process, the Heater, the two modes of the Valve, and the Controller, over which communication is enforced through the definitions $op = ro \, \gamma \, so$, $cl = rc \, \gamma \, sc$, and $H = \{so, sc, ro, rc\}$. In the next section, we will discuss an axiomatization of HyPA that allows us to rewrite this example into a form in which all parallel compositions are eliminated.

$$
\begin{array}{rcl}
\text{Water} & \approx & \left[\, w \mid \dot{w} = v - s \,\right] \\
\text{Heater} & \approx & \left[\, s_{min} \leq s \leq s_{max} \,\right] \\
\text{ValveOpen} & \approx & \left[\, v = v_{in} \,\right] \blacktriangleright rc \odot \text{ValveClose} \\
\text{ValveClose} & \approx & \left[\, v = 0 \,\right] \blacktriangleright ro \odot \text{ValveOpen} \\
\text{Controller} & \approx & \left[\, t \mid t^{+} = 0 \,\right] \gg \left[\, t \,\middle|\, \begin{array}{l} \dot{t} = 1 \\ t \leq T \end{array} \right] \blacktriangleright \left[\, t^{-} = T \,\right] \gg \\
 & & \left(\begin{array}{l} \left[\, w^{-} \geq w_{max} - w_{safe} \,\right] \gg sc \odot \text{Controller} \oplus \\ \left[\, w_{min} + w_{safe} \leq w^{-} \leq w_{max} - w_{safe} \,\right] \gg \text{Controller} \oplus \\ \left[\, w^{-} \leq w_{min} + w_{safe} \,\right] \gg so \odot \text{Controller} \end{array} \right) \\
\text{Boiler} & \approx & \partial_H \left(\text{Water} \parallel \text{Heater} \parallel (\text{ValveOpen} \oplus \text{ValveClosed}) \parallel \text{Controller} \right)
\end{array}
$$

3 Algebraic Reasoning in HyPA

The strength of the field of process algebra, lies in its ability to use equational reasoning for the analysis of transition systems, or, more precisely, for the analysis of equivalence classes of transition systems, called processes.

In this section, we show that this equational reasoning is also possible in HyPA. A notion of equivalence is defined on process terms, reflecting equivalence of the underlying semantical transition systems. Consequently, equivalent process terms represent the same process. We study properties of this equivalence, and capture those properties in a set of *derivation rules* and a set of *axioms* on the algebra of process terms. Together, this forms a proof system in which every derived equality on process terms represents equality of the underlying hybrid transition systems. In other words, process terms that are derivably equal, describe transition systems in the same equivalence class, and hence describe the same process.

This section is split up in three parts. In the first part, we define the well known notion of bisimulation equivalence on hybrid transition systems, we give a formal axiomatization, and prove soundness of this axiomatization. In the second part, we will treat the intuition behind the axioms, and insights they provide us with. In the third part, we show a few useful properties of our axiomatization, like a conservativity theorem with respect to the process algebra ACP and a rewrite system for rewriting closed HyPA terms into a normal form.

3.1 Axiomatization

The equivalence we assume on hybrid transition systems, is the well known notion of bisimulation [29].

Definition 7 (Bisimulation) *A relation $\mathcal{R} \subseteq P \times P$ on process terms, is a* bisimulation *relation if for all $p, q \in P$ such that $p \,\mathcal{R}\, q$, and for all valuations $\nu, \nu' \in Val$ and labels $l \in \mathbb{A} \cup \Sigma$, we find*

- $\langle p, \nu \rangle \checkmark$ *implies* $\langle q, \nu \rangle \checkmark$;
- $\langle q, \nu \rangle \checkmark$ *implies* $\langle p, \nu \rangle \checkmark$;
- $\langle p, \nu \rangle \xrightarrow{l} \langle p', \nu' \rangle$ *implies there exists* q' *s.t.* $\langle q, \nu \rangle \xrightarrow{l} \langle q', \nu' \rangle$ *and* $p' \mathcal{R} q'$;
- $\langle q, \nu \rangle \xrightarrow{l} \langle q', \nu' \rangle$ *implies there exists* p' *s.t.* $\langle p, \nu \rangle \xrightarrow{l} \langle p', \nu' \rangle$ *and* $p' \mathcal{R} q'$.

Two process terms x and y are bisimilar, *denoted $x \leftrightarrows y$, if there exists a bisimulation relation that relates them.*

If two process terms are bisimilar, then they describe equivalent transition systems, hence they describe the same process. In table 3 we give a set of derivation rules, and throughout the next subsection we give a set of axioms that, to a large extend, capture this notion of bisimulation. We write HyPA $\vdash p \approx q$, if we can derive equivalence of p and q using those axioms.

Definition 8 (Derivation) *We write HyPA $\vdash p \approx q$ to indicate that equivalence of open terms p and q can be derived from our axiom system. We define that equivalence can be derived according to the rules given in table 3.*

$$\frac{}{\text{HyPA} \vdash p \approx p}(1) \quad \frac{\text{HyPA} \vdash p \approx q}{\text{HyPA} \vdash q \approx p}(2) \quad \frac{\text{HyPA} \vdash p \approx q,\ \text{HyPA} \vdash q \approx r}{\text{HyPA} \vdash p \approx r}(3)$$

$$\frac{\text{HyPA} \vdash p \approx q,\ S \text{ a variable substitution}}{\text{HyPA} \vdash S(p) \approx S(q)}(4)$$

$$\frac{\mathcal{O} \text{ an n-ary HyPA operator, } \forall_{1 \leq i \leq n} \text{ HyPA} \vdash p_i \approx q_i}{\text{HyPA} \vdash \mathcal{O}(p_1 \ldots p_n) \approx \mathcal{O}(q_1 \ldots q_n)}(5)$$

$$\frac{\forall_{\nu,\nu'}\ (\nu,\nu') \models d \text{ iff } (\nu,\nu') \models d'}{\text{HyPA} \vdash d \gg x \approx d' \gg x}(6) \quad \frac{\forall_{\nu,\sigma}\ (\nu,\sigma) \models c \text{ iff } (\nu,\sigma) \models c'}{\text{HyPA} \vdash c \approx c'}(7)$$

$$\frac{p \approx q \text{ is an axiom}}{\text{HyPA} \vdash p \approx q}(8)$$

$$\frac{\begin{array}{c}\forall_{\nu,\sigma}\ (\nu,\sigma) \models c' \text{ implies } (\nu,\sigma) \models c \\ \forall_{\nu,\nu',\sigma}\ (\nu,\nu') \models d \text{ and } (\nu',\sigma) \models c \text{ implies } (\nu',\sigma) \models c'\end{array}}{\text{HyPA} \vdash d \gg c \approx d \gg c' \rhd c}(9)$$

$$\frac{\forall_{\nu,\sigma}\ (\nu,\sigma) \models c \text{ iff } (\nu,\sigma) \models c' \text{ or } (\nu,\sigma) \models c''}{\text{HyPA} \vdash c \approx (c' \oplus c'') \rhd c}(10)$$

Table 3: Derivation rules of HyPA

In the remainder of this subsection, the axioms of HyPA, and the insight they provide regarding the operators of the language, are presented. Also, the intuitions behind the 9th and 10th derivation rule, are discussed. In each of the axioms, x, y, z denote arbitrary open HyPA terms. The letters a, a' denote actions, while c, c' denote flow-clauses and d, d' denote reinitialization clauses. Unlike what is usual for ACP, one may not choose δ when a is written in an axiom. The set of axioms is divided into seven groups.

- The first group consists of two axioms that give the definition of parallel composition and the disrupt operator in terms of other HyPA operators.

  $$\begin{array}{c} x \parallel y \approx x \mathbin{\lfloor\!\lfloor} y \oplus y \mathbin{\lfloor\!\lfloor} x \oplus x \mid y \\ x \blacktriangleright y \approx x \rhd y \oplus y \end{array}$$

  These axioms can also be found in [8] and [4], respectively.

- The second group expresses associativity, commutativity and distribution properties of the various operators. All of these axioms occur also in [8] or [4]. Their intuition is the same as with standard process algebra.

  $$\begin{array}{cc} (x \oplus y) \oplus z \approx x \oplus (y \oplus z) & (x \odot y) \odot z \approx x \odot (y \odot z) \\ (x \rhd y) \rhd z \approx x \rhd (y \blacktriangleright z) & (x \mathbin{\lfloor\!\lfloor} y) \mathbin{\lfloor\!\lfloor} z \approx x \mathbin{\lfloor\!\lfloor} (y \parallel z) \\ (x \mid y) \mid z \approx x \mid (y \mid z) & (x \mid y) \mathbin{\lfloor\!\lfloor} z \approx x \mid (y \mathbin{\lfloor\!\lfloor} z) \\ (x \oplus y) \odot z \approx x \odot z \oplus y \odot z & (x \oplus y) \rhd z \approx x \rhd z \oplus y \rhd z \\ (x \oplus y) \mathbin{\lfloor\!\lfloor} z \approx x \mathbin{\lfloor\!\lfloor} z \oplus y \mathbin{\lfloor\!\lfloor} z & (x \oplus y) \mid z \approx x \mid z \oplus y \mid z \\ x \oplus y \approx y \oplus x & x \mid y \approx y \mid x \end{array}$$

  Notice, that these axioms may be used to prove the equalities $(x \blacktriangleright y) \blacktriangleright z \approx x \blacktriangleright (y \blacktriangleright z)$ and $(x \parallel y) \parallel z \approx x \parallel (y \parallel z)$.

- The third group is concerned with unit and zero elements for the various operators, and with axioms that express similar properties.

  $$\begin{array}{rr} x \odot \epsilon \approx x & x \rhd \delta \approx x \\ \epsilon \rhd x \approx \epsilon & \delta \mid x \approx \delta \\ \epsilon \parallel x \approx x & \epsilon \mathbin{\lfloor\!\lfloor} x \approx \delta \\ \left[\mathit{false}\right] \approx \delta & d \gg \delta \approx \delta \\ \left[\mathit{false}\right] \gg x \approx \delta & \left[\mathit{true}\right] \gg x \approx x \\ c_{jmp} \gg c \approx c & \end{array}$$

  Special attention should be paid to the axiom $c_{jmp} \gg c \approx c$, which expresses the intuition that every flow-clause may spontaneously reinitialize according the derived reinitialization-clause c_{jmp}. Furthermore, according to the axiom $d \gg (d' \gg x) \approx (d \sim d') \gg x$ found further on, we derive $c \approx c_{jmp} \gg c \approx c_{jmp} \gg c_{jmp} \gg c \approx (c_{jmp} \sim c_{jmp}) \gg c$, which expresses that multiple of those reinitializations may occur after each other.

- The fourth group of axioms focusses on the left-parallel composition and the left-disrupt operator.

$$
\begin{array}{ll}
(a \odot x) \lfloor\!\lfloor\, y \approx a \odot (x \,\|\, y) & (a \odot x) \rhd y \approx a \odot (x \blacktriangleright y) \\
(c \rhd x) \lfloor\!\lfloor\, y \approx \delta & (d \gg (c \rhd x)) \odot y \approx d \gg c \rhd (x \odot y)
\end{array}
$$

  The left-parallel composition operator is used to axiomatize the interleaving behaviour of the parallel composition. The axioms describe that only actions are allowed by left-parallel composition. The left process cannot perform signal transitions, since these should be synchronized. The left-disrupt operator is used to axiomatize the behaviour of a disrupt composition, when the left process is not disrupted immediately. Both actions and signals can be performed by the left-disrupt. However, the axiom expressing execution of a signal, i.e. $(d \gg c \rhd x) \odot y \approx d \gg c \rhd (x \odot y)$, is a little complicated because of the interaction between reinitialization-clauses and flow-clauses. Using the unit element $\left[\, \mathit{true} \,\right]$ for reinitialization we may clarify things a bit by obtaining the equality $(c \rhd x) \odot y \approx c \rhd (x \odot y)$. This equality is in itself not enough, because reinitialization does not distribute over sequential composition in a simple way, as we will see further on.

- The fifth group of axioms focusses on distribution properties of reinitializations.

$$
\begin{array}{ll}
d \gg x \oplus d' \gg x \approx (d \vee d') \gg x & (d \gg a) \odot x \approx d \gg a \odot x \\
d \gg (x \oplus y) \approx d \gg x \oplus d \gg y & (d \gg \epsilon) \odot x \approx d^{?} \gg x \\
(d \gg x) \rhd y \approx d \gg x \rhd y & (d \gg x) \lfloor\!\lfloor\, y \approx d \gg (x \lfloor\!\lfloor\, y) \\
d \gg (d' \gg x) \approx (d \sim d') \gg x &
\end{array}
$$

  A trivial consequence (using logical equivalence of $\left[\, \mathit{true} \,\right] \vee \left[\, \mathit{true} \,\right]$ and $\left[\, \mathit{true} \,\right]$) of these axioms is for example the equality $x \oplus x \approx x$, which expresses that the choice between equals is not an actual choice. Note, that reinitialization does not simply distribute over sequential composition! The reader should pay attention to the axiom $(d \gg \epsilon) \odot x \approx d^{?} \gg x$, which expresses that a reinitialization of the empty process only leads to termination if the reinitialization clause is satisfiable. I.e. only if there is a reinitialization possible that satisfies the clause. Note, that this reinitialization does not actually take place, therefore after termination the valuation of the variables is the same as before. Clearly, we can use the logical equivalence between $\left[\, \mathit{true} \,\right]$ and $\left[\, \mathit{true} \,\right]^{?}$ to obtain the equality $\epsilon \odot x \approx x$, known as an axiom from [8]. The last axiom in this group, expresses that a concatenation of reinitializations leads to a reinitialization with the concatenation of the clauses.

- The sixth group expresses the rather complicated rules for the synchronization operator. Since reinitialization does not distribute over synchronization, we have to take it into account in every of the axioms.

$$
\begin{array}{ll}
d \gg \epsilon \mid d' \gg \epsilon \approx (d^{?} \wedge d'^{?}) \gg \epsilon & \\
d \gg \epsilon \mid d' \gg (a \odot x) \approx \delta & \\
d \gg a \odot x \mid d' \gg a' \odot y \approx (d \wedge d') \gg (a\gamma a') \odot (x \,\|\, y) & \text{if } (a\gamma a') \text{ defined} \\
d \gg a \odot x \mid d' \gg a' \odot y \approx \delta & \text{if } (a\gamma a') \text{ undefined} \\
d \gg \epsilon \mid d' \gg c \rhd x \approx (d^{?} \sim d') \gg (c \rhd x) & \\
d \gg c \rhd x \mid d' \gg a \odot y \approx \delta &
\end{array}
$$

$$
\begin{array}{l}
d \gg c \rhd x \mid d' \gg c' \rhd y \approx \\
((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd \left(\begin{array}{l} x \,\lfloor\!\lfloor\, c' \blacktriangleright y \oplus \\ y \,\lfloor\!\lfloor\, c \blacktriangleright x \oplus \\ x \mid c' \blacktriangleright y \oplus \\ y \mid c \blacktriangleright x \end{array} \right)
\end{array}
$$

The first axiom expresses that a synchronization terminates if both the left and the right process terminate. Similar to the sequential composition, this termination only takes place if (both) the reinitializations are satisfiable. The second axiom expresses that termination cannot synchronize with actions, and therefore leads to deadlock. The third and fourth axiom express that actions a and a' may synchronize by producing an action $a\gamma a'$ if this action is defined, and otherwise the synchronization results in deadlock. If the synchronizing actions are reinitialized, both reinitializations should be satisfied, i.e. both processes should agree on the change of valuation. In particular, if $a\gamma a' = a''$, and a is reinitialized by an assignment $x^{+} = x^{-} + 1$, we find $[\, x \mid x^{+} = x^{-} + 1 \,] \gg a \mid a' \approx [\, x \mid x^{+} = x^{-} + 1 \,] \gg a \mid [\, \mathit{true} \,] \gg a' \approx ([\, x \mid x^{+} = x^{-} + 1 \,] \wedge [\, \mathit{true} \,]) \gg (a\gamma a') \approx [\, \mathit{false} \,] \gg (a\gamma a') \approx \delta$. Since a' does not agree on the assignment, a deadlock results. The fifth axiom expresses that termination may occur before signal behaviour executes. The terminating process disappears from the equation but, again, only if the corresponding reinitialization is satisfiable. That termination occurs *before* and not at *the same time* as the signal behaviour is expressed by the fact that we find a concatenation of reinitializations, rather than a conjunction. The sixth rule expresses that actions and signals cannot synchronize. Finally, the seventh axiom expresses the way in which signals can synchronize. This axiom is quite complicated due to our decision to make it possible for flow-clauses to perform reinitializations. When synchronizing, these flow-clause reinitializations should be taken into account. If we restrict ourselves to flow-clauses in which all variables are continuous (as is done in hybrid automata for example), i.e. clauses of the form $[\, \mathbb{V}_m \mid \mathrm{Pred} \,]$, we find the equality $d \gg c \rhd x \mid d' \gg c' \rhd y \approx (d \wedge d') \gg (c \wedge c') \rhd (x \,\|\, y)$, which is more in line with out intuition that both reinitialization-clauses and flow-clauses are synchronized. The proof of equality relies on the observation that, in case of continuity, $c_{jmp} = c^?_{jmp}$ (no jumps, hence only satisfiability) and $(d_0 \sim d^?_1) \wedge (d'_0 \sim d'^?_1) = (d_0 \wedge d'_0) \sim (d^?_1 \wedge d'^?_1)$.

- The seventh group expresses the (usual) distribution properties of the encapsulation.

$$\begin{array}{ll}
\partial_H(x \oplus y) \approx \partial_H(x) \oplus \partial_H(y) & \partial_H(x \odot y) \approx \partial_H(x) \odot \partial_H(y) \\
\partial_H(x \rhd y) \approx \partial_H(x) \rhd \partial_H(y) & \partial_H(d \gg x) \approx d \gg \partial_H(x) \\
\quad \partial_H(c) \approx c & \quad \partial_H(\epsilon) \approx \epsilon \\
\quad \partial_H(a) \approx a \ \text{ if } a \notin H & \quad \partial_H(a) \approx \delta \ \text{ if } a \in H
\end{array}$$

The 9th derivation rule in table 3, expresses how a reinitialization can restrict the choice for the first transition of a flow-clause. A useful application of this rule is in recognizing a solution of a differential equation given a certain initial condition. For example, consider the flow clause $\left[\, x, t \,\middle|\, \dot{x} = x \ \wedge \ \dot{t} = 1 \,\right]$. Clearly, $x = e^t$ is a solution of the differential equation $\dot{x} = x$, if initially $t = 0$ and $x = 1$. Using the 9th derivation rule, we now find the following equivalence.

$$\left[\, x, t \,\middle|\, \begin{matrix} x^+ = 1 \\ t^+ = 0 \end{matrix} \,\right] \gg \left[\, x, t \,\middle|\, \begin{matrix} \dot{x} = x \\ \dot{t} = 1 \end{matrix} \,\right]$$
$$\approx$$
$$\left[\, x, t \,\middle|\, \begin{matrix} x^+ = 1 \\ t^+ = 0 \end{matrix} \,\right] \gg \left[\, x, t \,\middle|\, \begin{matrix} x = e^t \\ \dot{t} = 1 \end{matrix} \,\right] \rhd \left[\, x, t \,\middle|\, \begin{matrix} \dot{x} = x \\ \dot{t} = 1 \end{matrix} \,\right] .$$

Note, that t and x are both taken to be continuous. Otherwise, the flow-clauses in this example might execute undesired reinitializations. The 9th derivation rule also expresses the repetitive character of flow-clauses. This is illustrated using $d = \left[\, \mathit{true} \,\right]$ and $c' = c$. We then find the equivalence $c \approx c \rhd c$.

The 10th derivation rule, also expresses this repetitive character. This is illustrated by taking $c = c' = c''$, we then find again $c \approx c \rhd c$. Furthermore, the 10th derivation rule expresses that if we can divide a flow-clause c into two (possibly overlapping) clauses c' and c'', then the first transition taken by c can be mimicked by either c' or c''. An application of this rule, is that a solution of a flow-clause can be split off even if there is no reinitialization. For example, the clause $\left[\, \begin{matrix} \dot{x} = 3x^{2/3} \\ \dot{t} = 1 \end{matrix} \,\right]$ contains a set of differential equations with solutions $x = 0$ and $x = t^3$ if initially $x = 0$ and $t = 0$. However, for other initial conditions, other solutions are possible. Using the 10th derivation rule, we find the following equality, which describes exactly that $x = 0$ and $x = t^3$ are two possible trajectories of this clause.

$$
\begin{gathered}
\begin{bmatrix} \dot{x} = 3x^{2/3} \\ \dot{t} = 1 \end{bmatrix} \\
\approx \\
\left(\begin{bmatrix} x = 0 \\ \dot{t} = 1 \end{bmatrix} \oplus \begin{bmatrix} x = 3x^{2/3} \\ \dot{t} = 1 \end{bmatrix} \right) \rhd \begin{bmatrix} x = 3x^{2/3} \\ \dot{t} = 1 \end{bmatrix} \\
\approx \\
\left(\begin{bmatrix} x = 0 \\ \dot{t} = 1 \end{bmatrix} \oplus \left(\begin{bmatrix} x = t^3 \\ \dot{t} = 1 \end{bmatrix} \oplus \begin{bmatrix} x = 3x^{2/3} \\ \dot{t} = 1 \end{bmatrix} \right) \rhd \begin{bmatrix} x = 3x^{2/3} \\ \dot{t} = 1 \end{bmatrix} \right) \rhd \begin{bmatrix} x = 3x^{2/3} \\ \dot{t} = 1 \end{bmatrix} \\
\approx \\
\begin{bmatrix} x = 0 \\ \dot{t} = 1 \end{bmatrix} \rhd \begin{bmatrix} x = 3x^{2/3} \\ \dot{t} = 1 \end{bmatrix} \oplus \left(\begin{bmatrix} x = t^3 \\ \dot{t} = 1 \end{bmatrix} \rhd \begin{bmatrix} x = 3x^{2/3} \\ \dot{t} = 1 \end{bmatrix} \right) \rhd \begin{bmatrix} x = 3x^{2/3} \\ \dot{t} = 1 \end{bmatrix} \oplus \\
\left(\begin{bmatrix} x = 3x^{2/3} \\ \dot{t} = 1 \end{bmatrix} \rhd \begin{bmatrix} x = 3x^{2/3} \\ \dot{t} = 1 \end{bmatrix} \right) \rhd \begin{bmatrix} x = 3x^{2/3} \\ \dot{t} = 1 \end{bmatrix} \\
\approx \\
\begin{bmatrix} x = 0 \\ \dot{t} = 1 \end{bmatrix} \rhd \begin{bmatrix} x = 3x^{2/3} \\ \dot{t} = 1 \end{bmatrix} \oplus \begin{bmatrix} x = t^3 \\ \dot{t} = 1 \end{bmatrix} \rhd \begin{bmatrix} x = 3x^{2/3} \\ \dot{t} = 1 \end{bmatrix} \oplus \begin{bmatrix} x = 3x^{2/3} \\ \dot{t} = 1 \end{bmatrix} \\
\approx \\
\left(\begin{bmatrix} x = 0 \\ \dot{t} = 1 \end{bmatrix} \oplus \begin{bmatrix} x = t^3 \\ \dot{t} = 1 \end{bmatrix} \right) \blacktriangleright \begin{bmatrix} x = 3x^{2/3} \\ \dot{t} = 1 \end{bmatrix}
\end{gathered}
$$

Note that, in contrast to the example for derivation rule 9, we do not need continuity of x and t in this case.

3.2 Soundness

Rests us to show, that all the derivations that can be made about process terms, indeed lead to sound statements about the bisimulation equivalence of these terms. In other words, we need to prove the following theorem.

Theorem 1 (Soundness) *If, for two closed terms p and q, we find HyPA $\vdash p \approx q$ then $p \leftrightarroweq q$.*

Proof Most of the techniques used in this proof are also explained in detail in [7]. The main observation is that a derivation is sound, if all of the rules that are used in it are sound. Now, if we use $\langle p \rangle \overset{\nu, l, \nu'}{\rightarrow} \langle p' \rangle$ as alternative notation for a transition $\langle p, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, then the definition of bisimulation becomes that of strong-bisimulation as defined in [7]. In other words, we use the obvious isomorphism, mapping $(P \times \mathit{Val}) \times (\mathbb{A} \cup \Sigma) \times (P \times \mathit{Val})$ to $P \times (\mathit{Val} \times (\mathbb{A} \cup \Sigma) \times \mathit{Val}) \times P$, to transform our notion of bisimulation into strong-bisimulation. Hence, the derivation rules 1, 2 and 3 are sound, since strong-bisimulation is an equivalence (see [29, 7, 43]).

The proof that rule 4 and 5 are sound, is based on the observation that, using the same isomorphism, all the semantical rules of HyPA turn out to be in the so-called *path*-format [6]. That is, formally they are in *path*-format only, if we read for example the rule $\frac{(\nu,\nu') \models d,\ \langle p,\nu' \rangle \checkmark}{\langle d \gg p,\nu \rangle \checkmark}$ as a rule $\frac{\langle p,\nu' \rangle \checkmark}{\langle d \gg p,\nu \rangle \checkmark}$ that is only valid under the condition that $(\nu,\nu') \models d$. It is a standard result [7, 6], that strong-bisimulation is a congruence for operators that are defined using only rules in *path*-format. Hence, in any context, variables may be replaced by terms, and terms may be replaced by equivalent terms, which is exactly what rule 4 and 5 express.

Rules 6 and 7 are sound, because the transition systems generated for logically equivalent clauses are isomorphic (hence bisimilar). This is straightforward to verify. That rule 8 is sound for each of the axioms of HyPA, and that the rules 9 and 9 are sound, is proven in appendix *A*. ⊠

3.3 Conservativity and Rewriting

One of the things that can be concluded about HyPA, using the given axiomatization, is that it is a conservative extension of the process algebra ACP [8]. This illustrates that HyPA does not violate the general ideas behind this process algebra.

Theorem 2 (Conservativity) *HyPA is a conservative extension of ACP (except for notational differences $\oplus$ and $\odot$), meaning that for every two closed ACP terms p and q, we find that ACP $\vdash p \approx q$ if and only if HyPA $\vdash p \approx q$.*

Proof See appendix B. ⊠

Furthermore, like in ACP, it is possible to define a set of basic terms into which every HyPA process can be rewritten using the axioms. These basic terms clearly show that the parallel compositions can be eliminated from all HyPA processes.

Definition 9 (Basic terms) *A basic term, is a process term of the following form. $N ::= d \gg \epsilon \mid d \gg a \odot N \mid d \gg c \rhd N \mid N \oplus N$.*

Theorem 3 (Rewriting) *Every closed HyPA term can be rewritten into a basic term.*

Proof In appendix C, we give a strongly normalizing rewrite system that does this, based (in principle) on reading all the axioms as rewrite rules from left to right (adding a few extra rules for handling unit elements). ⊠

We conjecture that this rewriting result can be extended to a linearization result, meaning that we expect to be able to rewrite every guarded recursive specification of a HyPA process into a linear form in which we only use recursion over basic terms.

The usefulness of elimination of the parallel composition, was already noted in the introduction. It was pointed out there, that the notion of bisimulation we use is very strong, because all possible valuations of the variables are taken into account at every point in time. Many weaker notions of equivalence, while still preserving interesting analysis properties, are not sensitive to the valuation of variables. Those equivalences, often, are not congruent for the parallel composition operator. Therefore, algebraic reasoning about those notions in the context of parallel composition becomes difficult.

This is already a known phenomenon in process theory, and it is caused by the possibility of interference in the value of shared variables (see for example [30]). Many different solutions have been proposed, also in the field of hybrid systems. For example, in the hybrid automaton theory of [28], the authors propose a restriction (called compatibility of automata) on the systems that may be placed in parallel, to ensure that no interference occurs. This is a perfectly reasonable way of handling the problem, but it has the disadvantage that we have to add extra variables, if we want to model processes that intentionally interfere, like the control system shown in the introduction.

HyPA is, in principle, focussed on being general. We start out by using a very general parallel composition, that is defined for all possible processes, and necessarily end up with an equivalence that is very strong, but is at least a congruence for this composition. Now, the elimination result allows us to eliminate the parallel composition from the process description. And, after elimination, we can start to use algebraic reasoning on a weaker notion of equivalence to analyse the specific properties we are interested in. Admittedly, this method may turn out to be less practical than the road followed by [28], because the elimination of parallel compositions can become quite cumbersome. On the other hand, it may also be possible to formulate derivation rules for reasoning about weaker notions of equivalence, that express a kind of conditional congruence 'under compatibility'. In this way, other methods can be imported into HyPA.

As an example of rewriting into basic terms, we can rewrite the steam boiler system of the previous section into the following description, in which parallel composition and encapsulation are eliminated. Notice that this rewriting is done here over a recursive definition, hence is an example of linearization of such process descriptions. Looking at the axiomatization, one might expect that $d_0, \ldots, d_3$ would contain clauses of the form c_{jmp}, but those are eliminated using calculation on reinitialization clauses. Admittedly, performing the actual elimination by hand is very cumbersome, and leads to a very long calculation, which we left out of this report for reasons of space. Finding theorems to make these calculations shorter, is a topic for future research.

$$\begin{array}{rcl}
\text{Boiler} & \approx & \text{Open} \oplus \text{Closed} \\
\text{Open} & \approx & d_0 \gg c_o \blacktriangleright (d_1 \gg cl \odot \text{Closed} \oplus d_2 \gg \text{Open} \oplus d_3 \gg op \odot \text{Open}) \\
\text{Closed} & \approx & d_0 \gg c_c \blacktriangleright (d_1 \gg cl \odot \text{Closed} \oplus d_2 \gg \text{Closed} \oplus d_3 \gg op \odot \text{Open})
\end{array}$$

with

$$\begin{aligned}
d_0 &\equiv \left[\, t \;\middle|\; t^+ = 0 \,\right], \\
d_1 &\equiv \left[\, t^- = T \,\right] \;\wedge\; \left[\, w^- \geq w_{max} - w_{safe} \,\right], \\
d_2 &\equiv \left[\, t^- = T \,\right] \;\wedge\; \left[\, w_{min} + w_{safe} \leq w^- \leq w_{max} - w_{safe} \,\right], \\
d_3 &\equiv \left[\, t^- = T \,\right] \;\wedge\; \left[\, w^- \leq w_{min} + w_{safe} \,\right], \\
c_o &\equiv \left[\; t, w \;\middle|\; \begin{array}{l} \dot{t} = 1 \\ t \leq T \\ \dot{w} = v - s \\ s_{min} \leq s \leq s_{max} \\ v = v_{in} \end{array} \;\right]
\end{aligned}$$

and

$$c_c \equiv \left[\; t, w \;\middle|\; \begin{array}{l} \dot{t} = 1 \\ t \leq T \\ \dot{w} = v - s \\ s_{min} \leq s \leq s_{max} \\ v = 0 \end{array} \;\right].$$

One result that is missing, so far, is a proof that the given axiomatization is complete for bisimulation of HyPA terms. I.e. a proof that for closed terms p and q, if $p \leftrightarrows q$ then also HyPA $\vdash p \approx q$. We do not exclude the possibility yet, modulo completeness of the logical equivalence of flow-clauses and reinitialization-clauses, but the fact that the number of signals that is a solution of a flow-clause, and the number of valuation jumps that is a solution of a reinitialization-clause may be infinite, complicates matters seriously.

4 Related Work

In this section, we will compare HyPA, in an informal way, to hybrid formalisms that were previously developed.

4.1 Hybrid Automata

One of the most influential of all hybrid formalisms, is the hybrid automaton formalism described by Henzinger [26]. These automata consist of nodes in which certain differential equations are active under an invariant, and of guarded transitions between those nodes that model discrete actions. For example, the steam-boiler example (after rewriting it into a basic term) could be modelled as the hybrid automaton depicted in figure 7.

A general hybrid automaton is depicted in figure 8. Such an automaton is easily translated into a hybrid process algebraic term, using the following observations.

- The flow predicate P_{fx} in a node of an automaton, describes flows in a similar way as in HyPA. Only, in hybrid automata, all signals are continuous. Hence, we take $V = \mathbb{V}_m$ and find the clause $\left[\, \mathbb{V}_m \;\middle|\; P_{fx} \,\right]$. Note, that hybrid automata only allow differentiable solutions of flow predicates, while

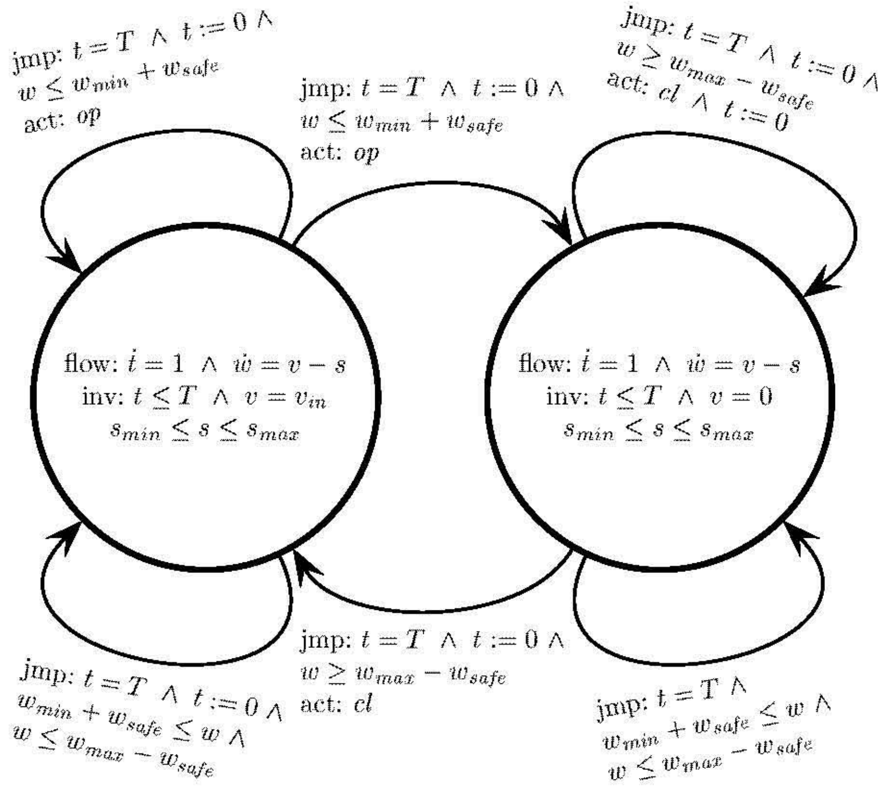
jmp: $t = T \wedge t := 0 \wedge$
$w \leq w_{min} + w_{safe}$
act: op
jmp: $t = T \wedge t := 0 \wedge$
$w \leq w_{min} + w_{safe}$
act: op
jmp: $t = T \wedge t := 0 \wedge$
$w \geq w_{max} - w_{safe}$
act: $cl \wedge t := 0$
flow: $\dot{t} = 1 \wedge \dot{w} = v - s$
inv: $t \leq T \wedge v = v_{in}$
$s_{min} \leq s \leq s_{max}$
flow: $\dot{t} = 1 \wedge \dot{w} = v - s$
inv: $t \leq T \wedge v = 0$
$s_{min} \leq s \leq s_{max}$
jmp: $t = T \wedge t := 0 \wedge$
$w_{min} + w_{safe} \leq w \wedge$
$w \leq w_{max} - w_{safe}$
jmp: $t = T \wedge t := 0 \wedge$
$w \geq w_{max} - w_{safe}$
act: cl
jmp: $t = T \wedge$
$w_{min} + w_{safe} \leq w \wedge$
$w \leq w_{max} - w_{safe}$

Figure 7: Example of a Hybrid Automaton Modelling a Steam Boiler

HyPA may allow non-differentiable solutions if a variable is not subject to differentiation. These additional solutions are considered to be unimportant for the moment. In a future, formal discussion of this translation, theory may be developed to handle them.

- The invariant P_{ix} is a predicate that can be used in a flow-clause, but can also be transformed to be used in a reinitialization clause, since only variables from the set $\mathbb{V}_m$ are used in it. The semantics of hybrid automata, contain a kind of look-ahead such that after a transition, an invariant P_{iy} or P_{iz} must hold respectively, other wise the transition cannot be taken. Translating this to HyPA, that means that in reinitializations, the predicate P_{iy}^{+} or P_{iz}^{+} should hold, respectively. Recall that we have defined P^{+} in section 2.2, as a transformation of a predicate P on $\mathbb{V}_m$ in which every variable x is replaced by x^{+}.

- The transitions of hybrid automata contain actions a_y and a_z. In translation, those actions disrupt the flow-clauses. Furthermore, the jump conditions P_{jy} and P_{jz} on the transitions are translated into reinitializations that act on these actions. Again, we take $V = \mathbb{V}_m$, and assume that it is specified in the jump condition which variables may change, and which remain constant.

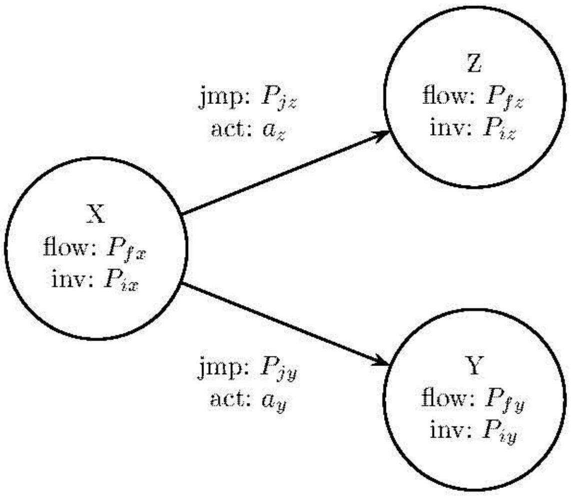

Figure 8: General Example of a Hybrid Automaton

Using these observations, the more general automaton in figure 8 is translated into:

$$X \approx [\, \mathbb{V}_m \mid P_{fx} \wedge P_{ix} \,] \blacktriangleright \left(\begin{array}{l} [\, \mathbb{V}_m \mid P_{jy} \wedge P_{iy}^{+} \,] \gg a_y \odot Y \\ \oplus \\ [\, \mathbb{V}_m \mid P_{jz} \wedge P_{iz}^{+} \,] \gg a_z \odot Z \end{array} \right).$$

Of course, this is not a formal translation. The semantics of hybrid automata as given in [26] is one of timed transition systems, while the hybrid transition systems we use here are subtly different. We conjecture that it is possible to transform the signal-transitions of the hybrid transition system into timed transitions, and the action-transitions of the hybrid transition system into action-transitions of a timed transition system, by abstracting away from all valuations. However, this is left as a subject for future research. The comparison with hybrid automata is merely intended to give an intuition on how the existing hybrid theories fit into our hybrid process algebraic structure.

4.2 Other Process Algebras

With respect to process algebras for hybrid systems, there are two previous works that we must consider. One, hybrid CSP, was already introduced in 1994 by Jifeng [27]. The other, ϕ-calculus, was very recently introduced by Rounds and Song in [35].

Hybrid CSP has a semantics in which each process represents a set of hybrid traces. Such a hybrid trace, then consists of a function of a continuous closed time domain to valuations, a function of that same domain to sequences (that gives the empty sequence except for on a finite set of time-points), and a few predicates (like termination). A system is then modelled in hybrid CSP, by giving a predicate that defines which traces are in the system. Comparable to the way that HyPA has atomic processes and operators, hybrid CSP has atomic predicates, and predicate operators. Apart from the fact that a trace semantics does not respect branching properties of a system, hybrid CSP also has the drawback that in parallel composition the continuous variables of the composed systems are assumed to be disjoint, and that assignments can only be made to programming variables, and not to continuous variables. We suspect, however, that these problems can be solved by defining new predicate operators, and that the author of [27] did not see the need for them at the time. Interestingly, there are operators defined in [27] whose function is not easily translated into HyPA. The main reason for this, is that clocks need to be modelled explicitly in HyPA, while they are often a functional part of the operators of hybrid CSP. Again, we conjecture, that HyPA can be extended with operators that mimic those of hybrid CSP, should the need arise.

The ϕ-calculus has a semantics based on timed transition systems, and given this, has a very interesting way of dealing with parallelism. As we already mentioned in the introduction, ϕ-calculus regards continuous behaviour to be a property of the environment, rather than a property of the ϕ-calculus program. Execution starts with an empty environment and, while running the program, differential equations (or rather their vector-field equivalents) and invariants, are added and replaced, by (interleavingly) executing so-called environmental actions. The upshot of this, is that it is not necessary to require that parallel programs have distinct continuous variables, but still, the semantics of the parallel composition of ϕ-calculus does not coincide with our intuition that continuous behaviour should simply satisfy both processes. Furthermore, because a vector-field is used as a representation of differential equations in the environment, ϕ-calculus can only handle differential equations with unique solutions (hence, not for example the equation $\dot{x} = 3x^{\frac{2}{3}}$). Also, the notion of equivalence that arises from using bisimulation in combination with environmental actions, makes that only syntactically equal differential equations are actually considered equal. This is a drawback that might be solved by some kind of abstraction, but it still has an artificial feel to it. Comparing ϕ-calculus to HyPA, we may conclude that, due to (amongst others) the environmental action approach, not all HyPA processes can be translated into ϕ-calculus. Conversely, the fact that the environmental actions of ϕ-calculus have a maximal progress semantics, ϕ-calculus programs cannot be translated into HyPA. This, however, can be solved by extending HyPA with an urgency operator, as was done for hybrid χ in [12, 36]

As we mentioned already in the introduction, HyPA is developed in close cooperation with the researchers developing hybrid χ. Research on the language χ, as a modelling and simulation language for process control, started in 1982

[34], and has since been through many stages of development, including an extension with hybrid description constructs. In 2002 [12], a formal operational semantics, based on CSP rather than ACP, was defined for the discrete-time part of the language, and recently, a formal semantics has been given for the hybrid part as well [36]. It is interesting to see that many of the theoretical aspects of HyPA (like the use of hybrid transition systems), have been applied in the formal semantics of χ, while on the other hand, the future extensions of HyPA are very likely to be inspired by the modelling strengths of χ, including their abstraction operators and possible the maximal progress operator. As research progressed, both languages seem to have evolved more and more towards each other, and it is not unthinkable that these paths will ultimately converge.

Another hybrid process algebra, was published as a technical report only one week before this one. In [11] a combination of the process algebra with continuous relative timing of [5] and the process algebra with propositional signals of [3], lead to a (only subtly) different algebra, that is also suited for the description of hybrid systems. The development of this algebra and of HyPA has been largely independent, and it is surprising to see how many similarities exist between the two. Nevertheless, due to different starting points and intuitions, also some differences can be found.

The process algebra of [11] was intended to be a conservative extension of timed ACP, while HyPA was intended to be an extension of 'normal' ACP. This gave rise to the most important difference, in our opinion, between the two languages, which is that [11] choice time-determinism (as it was discussed in section 2.2), while we chose time-non-determinism (which is more in line with the hybrid automaton approach [26]). As a matter of fact, in χ, two choice operators exists, one for each view on time. Another difference is that [11] intended to give an algebraic theory of hybrid automata, which leads to the modelling choice that switching between continuous behaviors can only take place through the use of discrete actions, while in HyPA switching can be arbitrary. This is illustrated, by the fact that the passing of time during which physical behaviour takes place, is modelled explicitly in [11], while, for HyPA, time passing is implicit when writing down a flow-clause.

4.3 Control Theory Formalisms

The formalisms used in control theory to describe hybrid systems can, from a HyPA point of view, be classified into two kinds. The first kind, are formalism regarding continuous time behaviour, while the second kind, regards time to evolve discretely. Roughly speaking, continuous time models can be translated into HyPA using flow-clauses, while the discrete models can be translated into reinitialization clauses, acting on a "time-step" process. Computational actions and sequential compositions of processes, seldomly play a role in control theory. Mode-switching, on the other hand, is a central aspect. In this paragraph, we sketch the general translation of several control theory formalisms into HyPA. We do not intend to be complete, but rather want to give a feel for the relation between HyPA and control theory. Furthermore, one has to keep in mind that control theory usually reasons about trace equivalence of systems, while HyPA is primarily concerned with bisimulation.

With respect to the continuous time models, we conjecture that most of them can be translated into either one singe flow-clause c or, in more complicated cases, into one single recursive term of the form:

$$\mathrm{CT} \approx (c_0 \oplus \ldots \oplus c_n) \blacktriangleright \mathrm{CT},$$

where $c_0 \ldots c_n$, denote clauses representing the different continuous modes a system can be in. If (and only if) a system can be modelled using only three continuous variables, namely the state variable $x \in \mathbb{R}^l$, the output variable $y \in \mathbb{R}^m$ and the input variable $u \in \mathbb{R}^n$, and using only clauses of the form

$$c_i = \left[\begin{array}{c|l} x & \dot{x} = A_i x + B_i u + f_i \\ & y = C_i x + D_i u + g_i \\ & (x,u) \in H_i \end{array}\right],$$

with A_i,B_i,C_i and D_i matrices of appropriate dimensions, and H_i a convex polyhedron (i.e. constructed from a finite set of inequalities), for every i, then we say that CT is a *continuous time piecewise affine* system [18]. If (and only if) a system can be modelled as one single continuous flow-clause, using the variables $v, w \in \mathbb{R}^s$ in addition to x,y and u, and if this flow clause is of the form

$$c = \left[\begin{array}{c|l} x & \dot{x} = Ax + B_1 u + B_2 w \\ & y = Cx + D_1 u + D_2 w \\ & v = E_1 x + E_2 u + E_3 w + e_4 \\ & 0 \leq v \perp w \geq 0 \end{array}\right]$$

then we say that the system is a *continuous time linear complementarity* system [40]. Here, $A, B_1, B_2, C, D_1, D_2, E_1, E_2$ and E_3 are matrices of appropriate dimensions, e_4 is a constant vector and $0 \leq v \perp w \geq 0$ denotes that the vectors v and w are orthogonal (i.e. $0 \leq v$, $0 \leq w$ and $v^T w = 0$).

A class of continuous control systems that does not fit directly into HyPA, is described by Filippov in [20]. The way in which he defines the solutions of differential inclusions, by using integration rather than derivation, is not captured by the notion of solution of a flow-clause in HyPA. Some differential inclusions do have solutions in Filippov's formalism, while they do not in HyPA. Filippov developed these kind of solutions, because the control community was struggling with a problem called "sliding modes" [41]. A particular example of this problem, is illustrated by the observation that in piece-wise affine systems, the system CT might get into deadlock on the borders of the polyhedra H_i, if the derivatives on both sides of the border "point" towards it. This deadlock is unintended, because the physical intuition of control scientists is usually that a system will start evolving along this border, rather than deadlocking. Filippov's method makes sure that these evolutions are included in the solutions of the differential inclusion describing the system. In HyPA, sliding modes will have to be modelled explicitly, by adding a separate clause c_s, for every polyhedron border where this phenomenon occurs. We expect, that Filippov's notion of solution can be adopted in the HyPA semantics, but do not know what the formal consequences would be precisely. This might be a subject of future research.

Discrete time models can be translated into the following HyPA term:

$$\mathrm{DT} \approx (d_0 \vee \ldots \vee d_n) \gg \mathrm{Timestep} \odot \mathrm{DT},$$

with

$$\mathrm{Timestep} \approx \left[\, t \,\middle|\, t^+ = 0 \right] \gg \left[\begin{array}{c|l} \{t\} \cup & \dot{t} = 1 \\ \{x_j \mid j \in J\} & t \leq T_s \\ & \bigwedge_{j \in J} \; \dot{x}_j = 0 \end{array} \right] \blacktriangleright \left[\, t^- = T_s \,\right] \gg \epsilon.$$

Here, the set $V = \{x_j \mid j \in J\}$ denotes the set of all variables that are used in the reinitialization-clauses $d_0 \ldots d_n$, describing the discontinuous changes over time. Timestep denotes the progress of time with one sample time $T_s > 0$, during which the variables x_j are supposed to remain constant. Similar to the continuous case, if (and only if) $V = \{x, y, u\}$, and for all reinitializations (with $i \in [0 \ldots n]$) we find

$$d_i = \left[\begin{array}{c|l} & x^+ = A_i x^- + B_i u^- + f_i \\ & y^+ = C_i x^+ + D_i u^+ + g_i \\ x, y, u & y^- = C_i x^- + D_i u^- + g_i \\ & (x^+, u^+) \in H_i \\ & (x^-, u^-) \in H_i \end{array} \right],$$

with A_i,B_i,C_i and D_i matrices of appropriate dimensions, and H_i a convex polyhedron, we say that *DT* is a *discrete time piecewise affine* system [18]. Analogously, if (and only if) a system can be written in the form

$$DT = \left[\begin{array}{c|l} & x^+ = A x^- + B_1 u^- + B_2 w^- \\ & y^+ = C x^+ + D_1 u^+ + D_2 w^+ \\ & y^- = C x^- + D_1 u^- + D_2 w^- \\ x, y, u & v^+ = E_1 x^+ + E_2 u^+ + E_3 w^+ + e_4 \\ v, w & v^- = E_1 x^- + E_2 u^- + E_3 w^- + e_4 \\ & 0 \leq v^+ \perp w^+ \geq 0 \\ & 0 \leq v^- \perp w^- \geq 0 \end{array} \right] \gg \mathrm{Timestep} \odot \mathrm{DT}\ ,$$

we say that it is a *discrete time linear complementarity* system [40].

A third type of discrete control formalism is *discrete time mixed logical dynamical* systems [10]. Similarly to linear complementarity systems, these systems can be described using only one reinitialization-clause. This time, however, the clause also reasons about variables that take value in the domain $\{0, 1\}$. A mixed logical dynamical system may use variables $x \in \mathbb{R}^l$, $y \in \mathbb{R}^m$ and $u \in \mathbb{R}^n$, and in addition, the variables $z \in \mathbb{R}^r$ and $w \in \{0, 1\}^s$, and can be written in the form:

$$DT = \left[\begin{array}{c|l} x, y, u & x^+ = Ax^- + B_1u^- + B_2w^- + B_3z^- \\ v, w & y^+ = Cx^+ + D_1u^+ + D_2w^+ + D_3z^+ \\ & y^- = Cx^- + D_1u^- + D_2w^- + D_3z^- \\ & E_1x^+ + E_2u^+ + E_3w^+ + E_4z^+ \leq e_5 \\ & E_1x^- + E_2u^- + E_3w^- + E_4z^- \leq e_5 \end{array} \right] \gg \text{Timestep} \odot \text{DT} \; .$$

In [25], the relation between the discrete control formalisms described above is further worked out, and it turns out that most of them are equivalent under certain, from a physical point of view very reasonable, assumptions.

As we mentioned in the beginning of this paragraphs, HyPA is primarily concerned with the notion of bisimulation equivalence. However, suppose we would adopt language equivalence, or even some weaker appropriate notion of equivalence. This would mean that we probably loose congruence of parallel composition, but it would also mean that we might be able to abstract away from a lot of computational behaviour and rewrite certain HyPA processes into one of the above forms. Since a lot of control theory is developed for those forms, this might greatly improve the analysis possibilities of HyPA.

5 Conclusions and Future Work

In this report, the syntax, semantics and axiomatization were presented, of a hybrid process algebraic theory called HyPA. This theory is aimed at the description and analysis of hybrid systems. HyPA is a conservative extension of the process algebra ACP [8], with a constant representing termination, a disrupt operator in the style of LOTOS [13], and clauses [41] for the description of continuous and discontinuous behaviour of model variables. Using the axiomatization of HyPA, closed process terms can be rewritten into basic terms, in which all parallel compositions are eliminated.

HyPA turns out to be different from most existing hybrid formalisms, in two major ways. It has a hybrid transition system semantics, for which it is not necessary to distinguish between state variables and external variables in differential equations. This allows for a general definition of parallel composition in the style of ACP, that also allows continuous interaction between all model variables. Furthermore, discontinuities in the variables of differential equations do not need to be explicitly modelled by assignment actions. Alternatively, in HyPA it is explicitly written down when a variable is continuous. Apparent drawbacks of HyPA are its strong notion of equivalence, and the sometimes complex axiomatization. However, we have sketched, how by assuming the same properties that are common on hybrid automata (compatibility of parallel composed systems, and continuity of all model variables), both the equivalence may be weakened, and the axiomatization becomes simpler. Admittedly, HyPA is very similar to the languages hybrid χ [36] and the hybrid process algebra of [11]. The differences are mainly found in the way time-determinism is treated, and in the way in which the passing of time is modelled implicitly or explicitly.

Future work on HyPA can be divided into five categories, given in arbitrary order.

- The first category, is a formalization of section 4, comparing HyPA to other (hybrid) formalisms. Clearly, since χ and the works of [11] are very similar, a formal comparison is indispensable. Also, formal comparisons with hybrid automata, ϕ-calculus, and hybrid Petri-nets, are important. Translations to and from those formalisms are useful, in order to be able to use analysis techniques from one, in the other formalism. This, of course, is also the case for various control formalisms and techniques.

- The second category, is the application of HyPA to a number of (larger) case studies. Only this will reveil whether the way of modelling we have chosen is indeed as convenient as expected, and whether practical theorems can be formulated to support the analysis of hybrid systems.

- The third category encompasses work on showing that the axiomatization of HyPA, modulo calculation on clauses, is complete (or can be made complete) for the notion of bisimulation. Also, extending the result for rewriting closed terms into basic terms, to rewriting of recursive specifications into a linear form, is essential for the analysis of systems.

- The fourth category of future work, is the extension of the theory with abstraction. Also, extension with system theoretic concepts like, for example, a metric or topology on the state-space [14], or other notions of limit behaviour [46], may then come into play. One of the classical problems in the hybrid systems field, namely the analysis of Zeno-behaviour, where infinite sequences of actions converge to a certain point, arises from such a metric, and we feel that a truly hybrid semantical model should include it. It is important to note, that without abstraction, our current notion of equivalence is strong enough to capture Zeno-behaviour, simply because process terms need to be equivalent for all valuations of variables, including Zeno-points. After abstraction of certain variables, however, Zeno-behaviour of those variables cannot be distinguished anymore, and therefore a new notion of equivalence might be needed. Other types of abstraction, like abstraction from actions [8, 21], would also greatly improve the analytic powers of HyPA. Also for those, new notions of bisimulation, known in classical process algebra for example, branching bisimulation, or observational equivalence, are needed.

- The fifth category, is tool support. Calculations on a simple example such as the steam-boiler, quickly become very cumbersome and tedious. This is a serious problem when applying the theory to any system of interesting size. Using the result that processes can be rewritten into basic terms using a strongly terminating rewriting system, makes that developing a very basic tool for partially automating these calculations should not be difficult.

Acknowledgements Finally, we would like to thank Paul van den Bosch, Jan Friso Groote, Maurice Heemels, Aleksandar Juloski, Tim Willemse, Ka Lok Man, Ramon Schiffelers, Bert van Beek and Mohammad Mousavi, for their comments during several stages of the development of this report.

References

[1] J-R. Abrial. Steam-boiler control specification problem. In *Dagstuhl Meeting: Methods for Semantics and Specification*, 1995.

[2] P. Amthor. A CSP model for hybrid automata. In *Northern Formal Methods Workshop (NFMW98)*, 1998.

[3] J.C.M. Baeten and J.A. Bergstra. Process algebra with propositional signals. *Theoretical Computer Science*, 177:381–405, 1997.

[4] J.C.M. Baeten and J.A. Bergstra. Mode transfer in process algebra. Technical Report CS-R 00-01, TU/e, 2000.

[5] J.C.M. Baeten and C.A. Middelburg. *Process Algebra with Timing*. Monographs in Theoretical Computer Science. Springer-Verlag, 2002.

[6] J.C.M. Baeten and C. Verhoef. A congruence theorem for structured operational semantics with predicates. In *Proceedings CONCUR '93*, volume 715 of *Lecture Notes in Computer Science*, pages 477–492. Springer-Verlag, 1993.

[7] J.C.M. Baeten and C. Verhoef. Concrete process algebra. In S. Abramsky, Dov M. Gabbay, and T.S.E. Maibaum, editors, *Semantic Modelling*, volume 4 of *Handbook of Logic in Computer Science*, pages 149–268. 1995.

[8] J.C.M. Baeten and W.P. Weijland. *Process Algebra*, volume 18 of *Cambridge Trancts in Theoretical Computer Science*. Cambridge University Press, Cambridge, 1990.

[9] J. Le Bail, H. Alla, and R. David. Hybrid Petri net. In *Proc. of the 1st European Control Conference, ECC'91*, pages 1472–7, Grenoble, France, July, 1991.

[10] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35.

[11] J.A. Bergstra and C.A. Middelburg. Process algebra for hybrid systems. Technical Report CSR 03-06, TU/e, Eindhoven, Netherlands, 2003.

[12] V. Bos and J.J.T. Kleijn. *Formal specification and analysis of industrial systems*. PhD thesis, TU/e, 2002.

[13] E. Brinksma. A tutorial on LOTOS. In Michel Diaz, editor, *Proc. Protocol Specification, Testing and Verification V*, pages 171–194, Amsterdam, Netherlands, 1985.

[14] P.J.L. Cuijpers and M.A. Reniers. Topological (bi-)simulation. Technical Report CS-Report 02-04, TU/e, Eindhoven, Netherlands, 2002.

[15] P.J.L. Cuijpers, M.A. Reniers, and W.P.M.H. Heemels. Hybrid transition systems. Technical Report CS-Report 02-12, TU/e, Eindhoven, Netherlands, 2002.

[16] I. Demongodin and N.T. Koussoulas. Differential Petri nets: A new model for hybrid systems. In *Proc. Advanced Summer Institute '96*, pages 61–8, June, 1996.

[17] R.C. Dorf and R.H. Bishop. *Modern Control Systems*. Series in Electrical and Computer Engineering: Control Engineering. Addison-Wesley, 1995.

[18] E.D.Sontag. Nonlinear regulation: The piecewise linear approach. *IEEE Trans. Autom. Control*, 26:346–358, 1981.

[19] A. Di Febbraro, A. Giua, and G. Menga, editors. *Special Issue on Hybrid Petri Nets*, volume 11 of *Discrete Event Dynamic Systems*, 2001.

[20] A.F. Filippov. *Differential Equations with Discontinuous Righthand Sides*. Mathematics and its applications (Soviet series). Kluwer Academic Press, 1988.

[21] W. Fokkink. *Introduction to Process Algebra*. Texts in Theoretical Computer Science. Springer-Verlag, Berlin, 1998.

[22] W. Fokkink, J.F. Groote, M. Hollenberg, and B. van Vlijmen. *LARIS 1.0: LAnguage for Railway Interlocking Specifications*. CWI, Amsterdam, 2000.

[23] J.F. Groote and M.A. Reniers. Algebraic process verification. In J.A. Bergstra, A. Ponse, and S.A. Smolka, editors, *Handbook of Process Algebra*, chapter 17, pages 1151–1208. Elsevier Science B.V., Amsterdam, 2001.

[24] J.F. Groote and J.J. van Wamel. Analysis of three hybrid systems in timed μCRL. *Science of Computer Programming*, 39:215–247, 2001.

[25] W.P.M.H. Heemels, B. De Schutter, and A. Bemporad. On the equivalence of classes of hybrid dynamical models. In *Proc. Conference on Decision and Control*, pages 364–369, Orlando, Florida, 2001.

[26] T.A. Henzinger. The theory of hybrid automata. In *Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science (LICS 1996)*, pages 278–292. IEEE Computer Society Press, 1996.

[27] H. Jifeng. From CSP to hybrid systems. In A.W.Roscoe, editor, *A Classical Mind, Essays in Honour of C.A.R. Hoare*, pages 171–189. Prentice-Hall International, 1994.

[28] N. Lynch, R. Segala, and F. Vaandrager. Hybrid I/O automata. *Information and Computation*.

[29] R. Milner. *A calculus of communicating systems*, volume 92 of *Lecture Notes in Computer Science*. Springer-Verlag, 1980.

[30] S.S. Owicki and D. Gries. An axiomatic proof technique for parallel programs I. *Acta Informatica*, 6:319–340, 1976.

[31] G.D. Plotkin. A structural approach to operational semantics. Technical Report DAIMI FN-19, Computer Science Department, Aarhus University, 1981.

[32] J.W. Polderman and J.C. Willems. *Introduction to Mathematical Systems Theory: A Behavioural Approach*, volume 26 of *Texts in Applied Mathematics*. Springer-Verlag, 1998.

[33] M. Rönkkö, A. P. Ravn, and K. Sere. Hybrid action systems. *Theoretical Computer Science*, 290:937–973, 2003.

[34] J.E. Rooda. *Simulation of Logistics Elements (Sole)*. Enschede, Netherlands, 1982. User Manual.

[35] W.C. Rounds and H. Song. The ϕ-calculus: A language for distributed control of reconfigurable embedded systems. In F. Wiedijk, O. Maler, and A. Pnueli, editors, *Hybrid Systems: Computation and Control, 6th International Workshop, HSCC 2003*, volume 2623 of *Lecture Notes in Computer Science*, pages 435–449. Springer-Verlag, 2003.

[36] R.R.H. Schiffelers, D.A. van Beek, K.L. Man, M.A. Reniers, and J.E. Rooda. Formal semantics of hybrid chi. In *Formats ?*, 2003.

[37] R.R.H. Schiffelers, D.A. van Beek, K.L. Man, M.A. Reniers, and J.E. Rooda. A hybrid language for modelling, simulation and verification. Analysis and Design of Hybrid Systems (ADHS03), to appear. International Federation of Automatic Control (IFAC), 2003.

[38] E.D. Sontag. *Mathematical Control Theory: Deterministic Finite Dimensional Systems*, volume 6 of *Texts in Applied Mathematics*. Springer-Verlag, 1998.

[39] D.A. van Beek, N.G. Jansen, K.L. Man, M.A. Reniers, J.E. Rooda, and R.R.H. Schiffelers. Relating chi to hybrid automata. In S.Chick, P.J. Sánchez, D. Ferrin, and D.J. Morrice, editors, *Proceedings of the 2003 Winter Simulation Conference*.

[40] A.J. van der Schaft and J.M. Schumacher. Complementarity modeling of hybrid systems. *IEEE Transactions on Automatic Control*, 43:483–490, 1998.

[41] A.J. van der Schaft and J.M. Schumacher. *An Introduction to Hybrid Dynamical Systems*, volume 251 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag, London, 2000.

[42] A.J. van der Schaft and J.M. Schumacher. Compositionality issues in discrete, continuous, and hybrid systems. *Int. J. Robust and Nonlinear Control*, 11:417–434, 2001.

[43] R.J. van Glabbeek. The linear time – branching time spectrum I: The semantics of concrete, sequential processes. In J.A. Bergstra, A. Ponse, and S.A. Smolka, editors, *Handbook of Process Algebra*, chapter 1, pages 3–99. Elsevier Science B.V., Amsterdam, 2001.

[44] J.J. Vereijken. A process algebra for hybrid systems. In *The Second European Workshop on Real-Time and Hybrid Systems*, Grenoble, France, 1995.

[45] T.A.C. Willemse. *Semantics and Verification in Process Algebras with Data and Timing*. PhD thesis, TU/e, Eindhoven, Netherlands, 2003.

[46] M. Ying. *Topology in Process Calculus: Approximate Correctness and Infinite Evolution of Concurrent Programs*. Springer-Verlag, 2001.

A Soundness

In this appendix, we prove soundness of derivation rules 9 and 10, and of all the axioms of HyPA. All proofs will be of the following form: (1) Every proof will have one subsection dedicated to it. (2) Every subsection starts out by giving a relation that is obviously a witness of the axiom or derivation rule under study. (3) The remainder of that section is devoted to proving that the given relation is a bisimulation relation, by verifying the truth of the four cases in the definition of bisimulation for every related pair.

A.1 Derivation Rule 9

Under the assumption that for all ν and σ we find that $(\nu, \sigma) \models c'$ implies $(\nu, \sigma) \models c$ and the assumption that for all ν, ν' and σ we find that $(\nu, \nu') \models d$ and $(\nu', \sigma) \models c$ implies $(\nu', \sigma) \models c'$, we study the smallest relation $\mathcal{R} \subseteq P \times P$ such that:

$$d \gg c \,\mathcal{R}\, d \gg c' \,\rhd\, c \;\wedge\; c \,\mathcal{R}\, c' \blacktriangleright c \;\wedge\; x \,\mathcal{R}\, x\ .$$

as a witness candidate for derivation rule 9.

For $x \,\mathcal{R}\, x$ the proof is trivial. For $d \gg c \,\mathcal{R}\, d \gg c' \rhd c$, it is easy to see that none of the related terms can terminate, hence we only need to check the following cases:

1. $\langle d \gg c, \nu \rangle \xrightarrow{l} \langle p, \nu'' \rangle$, for which we need the hypothesis
   (a) $\exists_{\nu'}\ (\nu, \nu') \models d$ and $\langle c, \nu' \rangle \xrightarrow{l} \langle p, \nu'' \rangle$, for which we need the hypothesis
      i. $\exists_{\sigma,t}\, p = c \wedge l = \sigma \wedge dom(\sigma) = [0 \ldots t] \wedge (\nu', \sigma) \models c \wedge \nu'' = \sigma(t)$. Now, using the assumptions stated in the beginning of this section, we conclude that $(\nu', \sigma) \models c'$, hence $\langle d \gg c' \rhd c, \nu \rangle \xrightarrow{l} \langle c' \blacktriangleright c, \nu'' \rangle$ with $p = c \,\mathcal{R}\, c' \blacktriangleright c$.

2. $\langle d \gg c' \rhd c, \nu \rangle \xrightarrow{l} \langle p, \nu'' \rangle$, for which we need the hypothesis
   (a) $\exists_{\nu'}\ (\nu, \nu') \models d$ and $\langle c' \rhd c, \nu' \rangle \xrightarrow{l} \langle p, \nu'' \rangle$, for which we need the hypothesis
      i. $\exists_{r}\ p = r \blacktriangleright c \;\wedge\; \langle c', \nu' \rangle \xrightarrow{l} \langle r, \nu'' \rangle$, for which we need the hypothesis
         A. $\exists_{\sigma,t}\ r = c' \;\wedge\; l = \sigma \;\wedge\; dom(\sigma) = [0 \ldots t] \;\wedge\; (\nu', \sigma) \models c' \;\wedge\; \sigma(t) = \nu''$.
         Using the assumptions stated in the beginning of this section we conclude $(\nu', \sigma) \models c$ and hence $\langle d \gg c, \nu \rangle \xrightarrow{l} \langle c, \nu'' \rangle$ with $c \,\mathcal{R}\, c' \blacktriangleright c = p$.

For $c \,\mathcal{R}\, c' \blacktriangleright c$, the proof is similar to that in derivation rule 10. Note, that this proof relies on the assumption that for all ν and σ such that $(\nu, \sigma) \models c'$ we find also $(\nu, \sigma) \models c$.

A.2 Derivation Rule 10

Under the assumption that $(\nu, \sigma) \models c$ if and only if $(\nu, \sigma) \models c'$ or $(\nu, \sigma) \models c''$, we study the smallest relation $\mathcal{R} \subseteq P \times P$ such that:

$$c \mathcal{R} (c' \oplus c'') \rhd c \ \wedge \ c \mathcal{R} c' \blacktriangleright c \ \wedge \ c \mathcal{R} c'' \blacktriangleright c \ \wedge \ \ x \mathcal{R} x \ .$$

as a witness candidate for derivation rule 10.

For $x \mathcal{R} x$ the proof is trivial. For $c \mathcal{R} (c' \oplus c'') \rhd c$, and the other cases, it is easy to see that none of the related terms can terminate, hence we only need to check the following cases:

1. $\langle c, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, from which we directly conclude $\langle c' \blacktriangleright c, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ and $\langle c'' \blacktriangleright c, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \mathcal{R} p$, and furthermore need the hypothesis
   (a) $\exists_{\sigma,t} \ l = \sigma \ \wedge \ dom(\sigma) = [0 \ldots t] \ \wedge \ p = c \ \wedge \ \nu' = \sigma(t) \ \wedge \ \sigma \models c$, for which we need one of the hypotheses
      i. $\sigma \models c'$
      For which we conclude $\langle c', \nu \rangle \xrightarrow{l} \langle c', \nu' \rangle$ and hence $\langle (c \oplus c') \rhd c, \nu \rangle \xrightarrow{l} \langle c' \blacktriangleright c, \nu' \rangle$, with $p \mathcal{R} c' \blacktriangleright c$.
      ii. $\sigma \models c''$
      Which is similar to the previous case.

2. $\langle (c' \oplus c'') \rhd c, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need the hypothesis
   (a) $\exists_r \ p = r \blacktriangleright c \ \wedge \ \langle c' \oplus c'', \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$, for which we need one of the hypotheses
      i. $\langle c', \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$, for which we need the hypothesis
         A. $\exists_{\sigma,t} \ l = \sigma \ \wedge \ dom(\sigma) = [0 \ldots t] \ \wedge \ r = c' \ \wedge \ \sigma \models c'$
         From which we conclude that $\sigma \models c$ hence $\langle c, \nu \rangle \xrightarrow{l} \langle c, \nu' \rangle$ with $c \mathcal{R} c' \blacktriangleright c$.
      ii. $\langle c'', \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$, which is similar to the previous case.

3. $\langle c' \blacktriangleright c, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need one of the hypotheses
   (a) $\exists_r \ p = r \blacktriangleright c \wedge \langle c', \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$, for which we need the hypothesis
      i. $r = c'$, from which we conclude $p = c' \blacktriangleright c$ hence $c \mathcal{R} p$.
   (b) $\langle c, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we conclude $p \mathcal{R} p$.

4. $\langle c'' \blacktriangleright c, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, which is similar to the previous case.

A.3 The axiom: $[\mathit{false}] \approx \delta$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that:

$$[\mathit{false}]\ \mathcal{R}\, \delta\ .$$

Clearly, since for no pair $(\nu, \sigma) \in \mathit{Val} \times \Sigma$ we find $(\nu, \sigma) \models [\mathit{false}]$, neither $[\mathit{false}]$ nor δ terminate, or perform any transition. Hence, $\mathcal{R}$ is a bisimulation relation.

A.4 The axiom: $[\mathit{false}] \gg x \approx \delta$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x \in P$:

$$[\mathit{false}] \gg x\, \mathcal{R}\, \delta.$$

Clearly, since for no pair $(\nu, \nu') \in \mathit{Val} \times \mathit{Val}$ we find $(\nu, \nu') \models [\mathit{false}]$, neither $[\mathit{false}]\, x \gg$ nor δ terminate, or perform any transition. Hence, $\mathcal{R}$ is a bisimulation relation.

A.5 The axiom: $d \gg d' \gg x \approx (d \sim d') \gg x$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x \in P$:

$$d \gg d' \gg x\, \mathcal{R}\, (d \sim d') \gg x\ \wedge\ x\, \mathcal{R}\, x\ .$$

For $x\, \mathcal{R}\, x$ the proof is trivial. For $d \gg d' \gg x\, \mathcal{R}\, (d \sim d') \gg x$, we find the following cases:

1. $\langle\, d \gg d' \gg x, \nu\,\rangle \checkmark$, for which we need the hypothesis
   (a) $\exists_{\nu'}\ (\nu, \nu') \models d\ \wedge\ \langle\, d' \gg x, \nu'\,\rangle \checkmark$, for which we need the hypothesis
      i. $\exists_{\nu''}\ (\nu', \nu'') \models d'\ \wedge\ \langle\, x, \nu''\,\rangle \checkmark$
      From which we conclude $(\nu, \nu'') \models d \sim d'$ and $\langle\, (d \sim d') \gg x, \nu\,\rangle \checkmark$.
2. $\langle\, (d \sim d') \gg x, \nu\,\rangle \checkmark$, for which we need the hypothesis
   (a) $\exists_{\nu''}\ (\nu, \nu'') \models d \sim d'\ \wedge\ \langle\, x, \nu''\,\rangle \checkmark$, for which we need the hypothesis
      i. $\exists_{\nu'}\ (\nu, \nu') \models d\ \wedge\ (\nu', \nu'') \models d'$
      From which we conclude $\langle\, d' \gg x, \nu'\,\rangle \checkmark$ and hence $\langle\, d \gg d' \gg x, \nu\,\rangle \checkmark$
3. $\langle\, d \gg d' \gg x, \nu\,\rangle \xrightarrow{l} \langle\, p, \nu'''\,\rangle$, for which we need the hypothesis
   (a) $\exists_{\nu'}\ (\nu, \nu') \models d\ \wedge\ \langle\, d' \gg x, \nu'\,\rangle \xrightarrow{l} \langle\, p, \nu'''\,\rangle$, for which we need the hypothesis

i. $\exists_{\nu''}\ (\nu',\nu'') \models d' \ \wedge\ \langle x,\nu'' \rangle \xrightarrow{l} \langle p,\nu''' \rangle$
From which we conclude $(\nu,\nu'') \models d \sim d'$, hence $\langle (d \sim d') \gg x,\nu \rangle \xrightarrow{l} \langle p,\nu''' \rangle$ with $p\,\mathcal{R}\,p$

4. $\langle (d \sim d') \gg x,\nu \rangle \xrightarrow{l} \langle p,\nu''' \rangle$, for which we need the hypothesis

   (a) $\exists_{\nu''}\ (\nu,\nu'') \models d \sim d' \ \wedge\ \langle x,\nu'' \rangle \xrightarrow{l} \langle p,\nu''' \rangle$, for which we need the hypothesis

   i. $\exists_{\nu'}\ (\nu,\nu') \models d \ \wedge\ (\nu',\nu'') \models d'$
   From which we conclude $\langle d' \gg x,\nu' \rangle \xrightarrow{l} \langle p,\nu''' \rangle$, and hence $\langle d \gg d' \gg x,\nu \rangle \xrightarrow{l} \langle p,\nu''' \rangle$, with $p\,\mathcal{R}\,p$

A.6 The axiom: $d \gg \delta \approx \delta$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that:

$$d \gg \delta\,\mathcal{R}\,\delta\ .$$

Trivially, both $d \gg \delta$ and δ cannot perform any transitions, nor terminate. Hence $\mathcal{R}$ is a bisimulation relation.

A.7 The axiom: $[\,true\,] \gg x \approx x$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x \in P$:

$$[\,true\,] \gg x\,\mathcal{R}\,x \ \wedge\ x\,\mathcal{R}\,x.$$

For $x\,\mathcal{R}\,x$, the proof is trivial. For $[\,true\,] \gg x\,\mathcal{R}\,x$ we find the following cases.

1. $\langle [\,true\,] \gg x,\nu \rangle \checkmark$, for which we need the hypothesis

   (a) $\exists_{\nu'}\ (\nu,\nu') \models [\,true\,] \ \wedge\ \langle x,\nu' \rangle \checkmark$, for which we need the hypothesis

   i. $\exists_{\upsilon}\ (\nu,\upsilon) \models [\,true\,] \ \wedge\ \nu = \nu'$
   From which we directly conclude $\langle x,\nu \rangle \checkmark$

2. $\langle x,\nu \rangle \checkmark$
   And clearly $(\nu,\nu) \models [\,true\,]$ hence $\langle [\,true\,] \gg x,\nu \rangle \checkmark$.

3. $\langle [\,true\,] \gg x,\nu \rangle \xrightarrow{l} \langle p,\nu'' \rangle$, for which we need the hypothesis

   (a) $\exists_{\nu'}\ (\nu,\nu') \models [\,true\,] \ \wedge\ \langle x,\nu' \rangle \xrightarrow{l} \langle p,\nu'' \rangle$, for which we need the hypothesis

i. $\exists_\upsilon\ (\nu, \upsilon) \models [\,true\,] \ \wedge \ \nu = \nu'$
From which we conclude $\langle\, x, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$ with $p \,\mathcal{R}\, p$.

4. $\langle\, x, \nu \,\rangle \xrightarrow{l} \langle\, p', \nu'' \,\rangle$
For which we may immediately conclude that $(\nu, \nu) \models [\,true\,]$, hence $\langle\, [\,true\,] \gg x, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$ with $p \,\mathcal{R}\, p$.

A.8 The axiom: $c_{jmp} \gg c \approx c$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x \in P$:

$$c_{jmp} \gg c \,\mathcal{R}\, c \ \wedge \ x \,\mathcal{R}\, x.$$

For $x \,\mathcal{R}\, x$, the proof is trivial, but for the proof of $c_{jmp} \gg c \,\mathcal{R}\, c$ we first need the following lemmas on solutions of flow clauses.

Lemma 1 *If $(\nu, \sigma') \models c$ and $(\sigma'(0), \sigma) \models c$ then $(\nu, \sigma) \models c$.*

Proof With induction to the structure of flow-clauses, we find two cases.

- If $(\nu, \sigma') \models [\,V \mid \text{Pred}\,]$ and $(\sigma'(0), \sigma) \models [\,V \mid \text{Pred}\,]$ then we find for all $t \in dom(\sigma)$ that $(\sigma(t), \dot{\sigma}(t)) \models \text{Pred}$ and furthermore for all $x \in V$ we find $\nu(x) = \sigma'(0)(x) = \sigma(0)(x)$, hence $(\nu, \sigma) \models [\,V \mid \text{Pred}\,]$.
- If $(\nu, \sigma') \models c \wedge c'$ and $(\sigma'(0), \sigma) \models c \wedge c'$ then, $(\nu, \sigma') \models c$ and $(\sigma'(0), \sigma) \models c$, and $(\nu, \sigma') \models c$ and $(\sigma'(0), \sigma) \models c$. Hence, with induction to the structure of c and c', we find $(\nu, \sigma) \models c$ and $(\nu, \sigma) \models c'$, and finally $(\nu, \sigma) \models c \wedge c'$.

⊠

Lemma 2 *If $(\nu, \sigma) \models c$ then $(\sigma(0), \sigma) \models c$.*

Proof With induction to the structure of flow-clauses, we find two cases.

- If $(\nu, \sigma) \models [\,V \mid \text{Pred}\,]$ then we find for all $t \in dom(\sigma)$ that $(\sigma(t), \dot{\sigma}(t)) \models \text{Pred}$ and furthermore, trivially, for all $x \in V$ we find $\sigma(0)(x) = \sigma(0)(x)$, hence $(\sigma(0), \sigma) \models [\,V \mid \text{Pred}\,]$.
- If $(\nu, \sigma) \models c \wedge c'$ then $(\nu, \sigma) \models c$ and $(\nu, \sigma) \models c'$. Hence, with induction to the structure of c and c', we find $(\sigma(0), \sigma) \models c$ and $(\sigma(0), \sigma) \models c'$, and finally $(\sigma(0), \sigma) \models c \wedge c'$.

⊠

Now, we proceed with the proof that $\mathcal{R}$ is a bisimulation relation. Since both $c_{jmp} \gg c$ and c cannot terminate, we only have the following two cases.

1. $\langle c_{jmp} \gg c, \nu \rangle \xrightarrow{l} \langle p, \nu'' \rangle$, for which we need the hypothesis

   (a) $\exists_{\nu'} \ (\nu, \nu') \models c_{jmp} \ \wedge \ \langle c, \nu' \rangle \overset{l}{\rightsquigarrow} \langle p, \nu'' \rangle$, for which we need the hypothesis

      i. $\exists_{\sigma,t,\sigma'} \ l = \sigma \ \wedge dom(\sigma) = [0 \ldots t] \ \wedge \ p = c \ \wedge \ (\nu', \sigma) \models c \ \wedge \ \nu'' = \sigma(t) \ \wedge \ (\nu, \sigma') \models c \ \wedge \ \nu' = \sigma'(0)$
      Using the first lemma, we conclude $(\nu, \sigma) \models c$ and hence $\langle c, \nu \rangle \xrightarrow{\sigma} \langle c, \sigma(t) \rangle$, i.e. $\langle c, \nu \rangle \xrightarrow{l} \langle p, \nu'' \rangle$ with $p \, \mathcal{R} \, p$

2. $\langle c, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need the hypothesis

   (a) $\exists_{\sigma,t} \ l = \sigma \ \wedge \ dom(\sigma) = [0 \ldots t] \ \wedge \ (\nu, \sigma) \models c \ \wedge \ \nu' = \sigma(t) \ \wedge \ p = c$
   Using the second lemma, we directly conclude that $(\sigma(0), \sigma) \models c$ and hence $(\nu, \sigma(0)) \models c_{jmp}$. Finally, this leads to $\langle c_{jmp} \gg c, \nu \rangle \xrightarrow{\sigma} \langle c, \sigma(t) \rangle$, i.e. to $\langle c_{jmp} \gg c, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \, \mathcal{R} \, p$.

A.9 The axiom: $(x \oplus y) \oplus z \approx x \oplus (y \oplus z)$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y, z \in P$:

$$(x \oplus y) \oplus z \, \mathcal{R} \, x \oplus (y \oplus z) \ \wedge \ x \, \mathcal{R} \, x.$$

For $x \, \mathcal{R} \, x$, the proof is trivial. For $(x \oplus y) \oplus z \, \mathcal{R} \, x \oplus (y \oplus z)$ we find the following cases.

1. $\langle (x \oplus y) \oplus z, \nu \rangle \checkmark$, for which we need one of the hypotheses

   (a) $\langle x \oplus y, \nu \rangle \checkmark$, for which we need one of the hypotheses

      i. $\langle x, \nu \rangle \checkmark$
      From which we directly conclude $\langle x \oplus (y \oplus z), \nu \rangle \checkmark$.

      ii. $\langle y, \nu \rangle \checkmark$
      From which we directly conclude $\langle y \oplus z, \nu \rangle \checkmark$ and $\langle x \oplus (y \oplus z), \nu \rangle \checkmark$

   (b) $\langle z, \nu \rangle \checkmark$
   From which we directly conclude $\langle y \oplus z, \nu \rangle \checkmark$ and $\langle x \oplus (y \oplus z), \nu \rangle \checkmark$

2. $\langle x \oplus (y \oplus z), \nu \rangle \checkmark$, similar to the previous case.

3. $\langle (x \oplus y) \oplus z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need one of the hypotheses

   (a) $\langle x \oplus y, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need one of the hypotheses

      i. $\langle x, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$
      From which we directly conclude $\langle x \oplus (y \oplus z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \, \mathcal{R} \, p$

ii. $\langle y, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$
From which we conclude $\langle y \oplus z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, hence $\langle x \oplus (y \oplus z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$

(b) $\langle z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$
From which we conclude $\langle y \oplus z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, hence $\langle x \oplus (y \oplus z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$

4. $\langle x \oplus (y \oplus z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, similar to the previous case.

A.10 The axiom: $x \oplus y \approx y \oplus x$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y \in P$:

$$x \oplus y \,\mathcal{R}\, y \oplus x \;\wedge\; x \,\mathcal{R}\, x.$$

For $x \,\mathcal{R}\, x$, the proof is trivial. For $x \oplus y \,\mathcal{R}\, y \oplus x$ we find the following cases.

1. $\langle x \oplus y, \nu \rangle \checkmark$, for which we need one of the hypotheses

   (a) $\langle x, \nu \rangle \checkmark$
   From which we directly conclude $\langle y \oplus x, \nu \rangle \checkmark$

   (b) $\langle y, \nu \rangle \checkmark$
   From which we directly conclude $\langle y \oplus x, \nu \rangle \checkmark$

2. $\langle y \oplus x, \nu \rangle \checkmark$, symmetrical to the previous case.

3. $\langle x \oplus y, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need one of the hypotheses

   (a) $\langle x, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$
   From which we directly conclude $\langle y \oplus x, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.

   (b) $\langle y, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ From which we directly conclude $\langle y \oplus x, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.

4. $\langle y \oplus x, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, symmetrical to the previous case.

A.11 The axiom: $d \gg x \oplus d' \gg x \approx (d \vee d') \gg x$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x \in P$:

$$d \gg x \oplus d' \gg x \,\mathcal{R}\, (d \vee d') \gg x \;\wedge\; x \,\mathcal{R}\, x.$$

For $x \,\mathcal{R}\, x$, the proof is trivial. For $d \gg x \oplus d' \gg x \,\mathcal{R}\, (d \vee d') \gg x$, we find the following cases.

1. $\langle\, d \gg x \oplus d' \gg x, \nu \,\rangle \checkmark$, for which we need one of the hypotheses
   (a) $\langle\, d \gg x, \nu \,\rangle \checkmark$, for which we need the hypothesis
      i. $\exists_{\nu'} (\nu, \nu') \models d \;\wedge\; \langle\, x, \nu' \,\rangle \checkmark$
      From which we conclude $(\nu, \nu') \models (d \vee d')$, hence $\langle\, (d \vee d') \gg x, \nu' \,\rangle \checkmark$.
   (b) $\langle\, d' \gg x, \nu \,\rangle \checkmark$, similar to the previous case.
2. $\langle\, (d \vee d') \gg x, \nu \,\rangle \checkmark$, for which we need the hypothesis
   (a) $\exists_{\nu'} (\nu, \nu') \models (d \vee d') \;\wedge\; \langle\, x, \nu' \,\rangle \checkmark$, for which we need one of the hypotheses
      i. $(\nu, \nu') \models d$
      From which we directly conclude $\langle\, d \gg x, \nu \,\rangle \checkmark$ and hence $\langle\, d \gg x \oplus d' \gg x, \nu \,\rangle \checkmark$.
      ii. $(\nu, \nu') \models d'$, similar to the previous case.
3. $\langle\, d \gg x \oplus d' \gg x, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$, for which we need one of the hypotheses
   (a) $\langle\, d \gg x, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$, for which we need the hypothesis
      i. $\exists_{\nu'} (\nu, \nu') \models d \;\wedge\; \langle\, x, \nu' \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$
      From which we conclude $(\nu, \nu') \models (d \vee d')$ hence $\langle\, (d \vee d') \gg x, \nu' \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$ with $p \,\mathcal{R}\, p$.
   (b) $\langle\, d' \gg x, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$, similar to the previous case.
4. $\langle\, (d \vee d') \gg x, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$, for which we need the hypothesis.
   (a) $\exists_{\nu'} (\nu, \nu') \models (d \vee d') \;\wedge\; \langle\, x, \nu' \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$, for which we need one of the hypotheses
      i. $(\nu, \nu') \models d$
      From which we directly conclude $\langle\, d \gg x, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$ and hence $\langle\, d \gg x \oplus d' \gg x, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$ with $p \,\mathcal{R}\, p$.
      ii. $(\nu, \nu') \models d'$, similar to the previous case.

A.12 The axiom: $d \gg (x \oplus y) \approx d \gg x \oplus d \gg y$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y \in P$:

$$d \gg (x \oplus y) \,\mathcal{R}\, d \gg x \oplus d \gg y \;\wedge\; x \,\mathcal{R}\, x.$$

For $x \,\mathcal{R}\, x$, the proof is trivial. For $d \gg (x \oplus y) \,\mathcal{R}\, d \gg x \oplus d \gg y$, we find the following cases.

1. $\langle\, d \gg (x \oplus y), \nu \,\rangle \checkmark$, for which we need the hypothesis

(a) $\exists_{\nu'}\ (\nu,\nu') \models d\ \wedge\ \langle\, x \oplus y, \nu' \,\rangle \checkmark$, for which we need one of the hypotheses

   i. $\langle\, x, \nu' \,\rangle \checkmark$
      From which we conclude $\langle\, d \gg x, \nu \,\rangle \checkmark$ hence $\langle\, d \gg x \oplus d \gg y, \nu \,\rangle \checkmark$.

   ii. $\langle\, y, \nu' \,\rangle \checkmark$
      Which is similar to the previous case.

2. $\langle\, d \gg x \oplus d \gg y, \nu \,\rangle \checkmark$, for which we need one of the hypotheses

   (a) $\langle\, d \gg x, \nu \,\rangle \checkmark$, for which we need the hypothesis

      i. $\exists_{\nu'}\ (\nu,\nu') \models d\ \wedge\ \langle\, x, \nu' \,\rangle \checkmark$
         From which we conclude $\langle\, x \oplus y, \nu' \,\rangle \checkmark$, and hence $\langle\, d \gg (x \oplus y), \nu \,\rangle \checkmark$.

   (b) $\langle\, d \gg y, \nu \,\rangle \checkmark$
      Which is similar to the previous case.

3. $\langle\, d \gg (x \oplus y), \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$, for which we need the hypothesis

   (a) $\exists_{\nu'}\ (\nu,\nu') \models d\ \wedge\ \langle\, x \oplus y, \nu' \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$, for which we need one of the hypotheses

      i. $\langle\, x, \nu' \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$
         From which we conclude $\langle\, d \gg x, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$ hence $\langle\, d \gg x \oplus d \gg y, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$.

      ii. $\langle\, y, \nu' \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$
         Which is similar to the previous case.

4. $\langle\, d \gg x \oplus d \gg y, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$, for which we need one of the hypotheses

   (a) $\langle\, d \gg x, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$, for which we need the hypothesis

      i. $\exists_{\nu'}\ (\nu,\nu') \models d\ \wedge\ \langle\, x, \nu' \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$
         From which we conclude $\langle\, x \oplus y, \nu' \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$, and hence $\langle\, d \gg (x \oplus y), \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$.

   (b) $\langle\, d \gg y, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$
      Which is similar to the previous case.

A.13 The axiom: $(x \oplus y) \odot z \approx x \odot z \oplus y \odot z$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y, z \in P$:

$$(x \oplus y) \odot z\, \mathcal{R}\, x \odot z \oplus y \odot z\ \wedge\ x\, \mathcal{R}\, x.$$

For $x\, \mathcal{R}\, x$, the proof is trivial. For $(x \oplus y) \odot z\, \mathcal{R}\, x \odot z \oplus y \odot z$, we find the following cases.

1. $\langle (x \oplus y) \odot z, \nu \rangle \checkmark$, for which we need the hypothesis
   (a) $\langle x \oplus y, \nu \rangle \checkmark \ \wedge \ \langle z, \nu \rangle \checkmark$, for which we need one of the hypotheses
      i. $\langle x, \nu \rangle \checkmark$
      From which we conclude $\langle x \odot z, \nu \rangle \checkmark$ and hence $\langle x \odot z \oplus y \odot z \rangle \checkmark$.
      ii. $\langle y, \nu \rangle \checkmark$, which is similar to the previous case.
2. $\langle x \odot z \oplus y \odot z, \nu \rangle \checkmark$, which follows the reverse reasoning of the previous case.
3. $\langle (x \oplus y) \odot z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need one of the hypotheses
   (a) $\langle x \oplus y, \nu \rangle \checkmark \ \wedge \langle z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need one of the hypotheses
      i. $\langle x, \nu \rangle \checkmark$
      From which we conclude $\langle x \odot z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ and hence $\langle x \odot z \oplus y \odot z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, with $p \mathcal{R} p$.
      ii. $\langle y, \nu \rangle \checkmark$
      Which is similar to the previous case.
   (b) $\exists_r \ p = r \odot z \ \wedge \ \langle x \oplus y, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$, for which we need one of the following hypotheses
      i. $\langle x, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$
      From which we conclude $\langle x \odot z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ and hence $\langle x \odot z \oplus y \odot z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, with $p \mathcal{R} p$.
      ii. $\langle y, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$
      Which is similar to the previous case.
4. $\langle x \odot z \oplus y \odot z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, which follows the reverse reasoning of the previous case.

A.14 The axiom: $(x \odot y) \odot z \approx x \odot (y \odot z)$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y, z \in P$:

$$(x \odot y) \odot z \mathcal{R} x \odot (y \odot z) \ \wedge \ x \mathcal{R} x.$$

For $x \mathcal{R} x$, the proof is trivial. For $(x \odot y) \odot z \mathcal{R} x \odot (y \odot z)$, we find the following cases.

1. $\langle (x \odot y) \odot z, \nu \rangle \checkmark$, for which we need the hypothesis
   (a) $\langle x \odot y, \nu \rangle \checkmark \ \wedge \ \langle z, \nu \rangle \checkmark$, for which we need the hypothesis

      i. $\langle x, \nu \rangle \checkmark \ \wedge \ \langle y, \nu \rangle \checkmark$
      From which we readily conclude $\langle y \odot z, \nu \rangle \checkmark$ and $\langle x \odot (y \odot z), \nu \rangle \checkmark$.

2. $\langle x \odot (y \odot z), \nu \rangle \checkmark$, similar to the previous case.

3. $\langle (x \odot y) \odot z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need one of the hypotheses

   (a) $\langle x \odot y, \nu \rangle \checkmark \ \wedge \ \langle z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need the hypothesis
      i. $\langle x, \nu \rangle \checkmark \ \wedge \langle y, \nu \rangle \checkmark$
      From which we readily conclude $\langle y \odot z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ and $\langle x \odot (y \odot z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \mathcal{R} p$.

   (b) $\exists_r \ p = r \odot z \ \wedge \ \langle x \odot y, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$, for which we need one of the hypotheses
      i. $\langle x, \nu \rangle \checkmark \ \wedge \ \langle y, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$
      From which we readily conclude that $\langle y \odot z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, and hence $\langle x \odot (y \odot z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, with $p \mathcal{R} p$.
      ii. $\exists_s \ r = s \odot y \ \wedge \ \langle x, \nu \rangle \xrightarrow{l} \langle s, \nu' \rangle$
      From which we readily conclude that $p = (s \odot y) \odot z$ and $\langle x \odot (y \odot z), \nu \rangle \xrightarrow{l} \langle s \odot (y \odot z), \nu' \rangle$ with $p \mathcal{R}\, s \odot (y \odot z)$.

4. $\langle x \odot (y \odot z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, similar to the previous case.

A.15 The axiom: $x \odot \epsilon \approx x$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x \in P$:

$$x \odot \epsilon \, \mathcal{R} \, x \ .$$

We only need to verify the following cases.

1. $\langle x \odot \epsilon, \nu \rangle \checkmark$, for which we need the hypothesis

   (a) $\langle x, \nu \rangle \checkmark$

2. $\langle x, \nu \rangle \checkmark$
   Using the rule for $\langle \epsilon, \nu \rangle \checkmark$, we immediately conclude $\langle x \odot \epsilon, \nu \rangle \checkmark$.

3. $\langle x \odot \epsilon, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need the hypothesis

   (a) $\exists_{p'} \ p = p' \odot \epsilon \ \wedge \langle x, \nu \rangle \xrightarrow{l} \langle p', \nu' \rangle$
   And by construction $p' \mathcal{R} p$.

4. $\langle x, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$
   From which we conclude $\langle x \odot \epsilon, \nu \rangle \xrightarrow{l} \langle p \odot \epsilon, \nu' \rangle$

A.16 The axiom: $(d \gg a) \odot x \;\approx\; d \gg a \odot x$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x \in P$:

$$(d \gg a) \odot x \;\approx\; d \gg a \odot x \;\wedge\; x \,\mathcal{R}\, x\ .$$

For $x \,\mathcal{R}\, x$, the proof is trivial. For $(d \gg a) \odot x \;\mathcal{R}\; d \gg a \odot x$, we find there is no termination, since actions do not terminate immediately. This leaves us with the following cases.

1. $\langle\, (d \gg a) \odot x, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$, for which we need one of the hypotheses
   (a) $\langle\, d \gg a, \nu \,\rangle \checkmark$
   Which cannot occur, since there is no termination rule for a.
   (b) $\exists_r\ p = r \odot x \;\wedge\; \langle\, d \gg a, \nu \,\rangle \xrightarrow{l} \langle\, r, \nu'' \,\rangle$, for which we need the hypothesis
      i. $\exists_{\nu'} (\nu, \nu') \models d \;\wedge\; \langle\, a, \nu' \,\rangle \xrightarrow{l} \langle\, r, \nu'' \,\rangle$
      From which we conclude $\langle\, a \odot x, \nu' \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$ hence $\langle\, d \gg a \odot x, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$ with $p \,\mathcal{R}\, p$.
2. $\langle\, d \gg a \odot x, \nu' \,\rangle \xrightarrow{l} \langle\, p', \nu'' \,\rangle$, with reverse reasoning to the previous case.

A.17 The axiom: $(d \gg \epsilon) \odot x \;\approx\; d^? \gg x$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x \in P$:

$$(d \gg \epsilon) \odot x \;\mathcal{R}\; d^? \gg x \;\wedge\; x \,\mathcal{R}\, x\ .$$

For $x \,\mathcal{R}\, x$, the proof is trivial. For $(d \gg \epsilon) \odot x \;\mathcal{R}\; d^? \gg x$, we find the following cases.

1. $\langle\, (d \gg \epsilon) \odot x, \nu \,\rangle \checkmark$, for which we need the hypothesis
   (a) $\langle\, d \gg \epsilon, \nu \,\rangle \checkmark \;\wedge\; \langle\, x, \nu \,\rangle \checkmark$, for which we need the hypothesis
      i. $\exists_\upsilon (\nu, \upsilon) \models d$
      From which we conclude that $(\nu, \nu) \models d^?$ and hence $\langle\, d^? \gg x, \nu \,\rangle \checkmark$
2. $\langle\, d^? \gg x, \nu \,\rangle \checkmark$, for which we need the hypothesis
   (a) $\exists_{\nu'}\ (\nu, \nu') \models d^? \;\wedge\; \langle\, x, \nu' \,\rangle \checkmark$, for which we need the hypothesis
      i. $\exists_\upsilon\ (\nu, \upsilon) \models d \;\wedge\; \nu = \nu'$
      Hence, using $\langle\, \epsilon, \upsilon \,\rangle \checkmark$ we may derive $\langle\, d \gg \epsilon, \nu \,\rangle \checkmark$ and finally, using $\nu = \nu'$, we find $\langle\, (d \gg \epsilon) \odot x, \nu \,\rangle \checkmark$.
3. $\langle\, (d \gg \epsilon) \odot x, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$, for which we need one of the hypotheses

(a) $\langle\, d \gg \epsilon, \nu\,\rangle \checkmark \;\wedge\; \langle\, x, \nu\,\rangle \xrightarrow{l} \langle\, p, \nu''\,\rangle$, for which we need the hypothesis

   i. $\exists_{\upsilon}\ (\nu, \upsilon) \models d$
   From which we conclude $(\nu, \nu) \models d^?$ and hence $\langle\, d^? \gg x, \nu\,\rangle \xrightarrow{l} \langle\, p, \nu''\,\rangle$ with $p\,\mathcal{R}\,p$.

(b) $\exists_{r}\ \langle\, d \gg \epsilon, \nu\,\rangle \xrightarrow{l} \langle\, r, \nu''\,\rangle$
Which cannot be fulfilled since ϵ does not generate any transitions.

4. $\langle\, d^? \gg x, \nu\,\rangle \xrightarrow{l} \langle\, p, \nu''\,\rangle$, for which we need the hypothesis

   (a) $\exists_{\nu'}\ (\nu, \nu') \models d^? \;\wedge\; \langle\, x, \nu'\,\rangle \xrightarrow{l} \langle\, p, \nu''\,\rangle$, for which we need the hypothesis

      i. $\exists_{\upsilon}\ (\nu, \upsilon) \models d \;\wedge\; \nu = \nu'$
      Hence, using $\langle\, \epsilon, \upsilon\,\rangle \checkmark$ we may derive $\langle\, d \gg \epsilon, \nu\,\rangle \checkmark$, and finally, using $\nu = \nu'$, we find $\langle\, (d \gg \epsilon) \odot x, \nu\,\rangle \xrightarrow{l} \langle\, p, \nu''\,\rangle$ with $p\,\mathcal{R}\,p$.

A.18 The axiom: $x \blacktriangleright y \approx x \vartriangleright y \oplus y$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y \in P$:

$$x \blacktriangleright y\,\mathcal{R}\,x \vartriangleright y \oplus y \;\wedge\; x\,\mathcal{R}\,x\ .$$

For $x\,\mathcal{R}\,x$, the proof is trivial. For $x \blacktriangleright y\,\mathcal{R}\,x \vartriangleright y \oplus y$, we find the following cases.

1. $\langle\, x \blacktriangleright y, \nu\,\rangle \checkmark$, for which we need one of the hypotheses

   (a) $\langle\, x, \nu\,\rangle \checkmark$
   From which we conclude $\langle\, x \vartriangleright y, \nu\,\rangle \checkmark$ hence $\langle\, x \vartriangleright y \oplus y, \nu\,\rangle \checkmark$.

   (b) $\langle\, y, \nu\,\rangle \checkmark$
   From which we directly conclude $\langle\, x \vartriangleright y \oplus y, \nu\,\rangle \checkmark$.

2. $\langle\, x \vartriangleright y \oplus y, \nu\,\rangle \checkmark$, for which we need one of the hypotheses

   (a) $\langle\, x \vartriangleright y, \nu\,\rangle \checkmark$, for which we need the hypothesis

      i. $\langle\, x, \nu\,\rangle \checkmark$
      From which we directly conclude $\langle\, x \blacktriangleright y, \nu\,\rangle \checkmark$.

   (b) $\langle\, y, \nu\,\rangle \checkmark$
   From which we directly conclude $\langle\, x \blacktriangleright y, \nu\,\rangle \checkmark$.

3. $\langle\, x \blacktriangleright y, \nu\,\rangle \xrightarrow{l} \langle\, p, \nu'\,\rangle$, for which we need one of the hypotheses

   (a) $\langle\, y, \nu\,\rangle \xrightarrow{l} \langle\, p, \nu'\,\rangle$
   From which we directly conclude $\langle\, x \vartriangleright y \oplus y, \nu\,\rangle \xrightarrow{l} \langle\, p, \nu'\,\rangle$, with $p\,\mathcal{R}\,p$.

(b) $\exists_r\ p = r \blacktriangleright y \wedge \langle x, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$
From which we conclude $\langle x \rhd y, \nu \rangle \xrightarrow{l} \langle r \blacktriangleright y, \nu' \rangle$ and hence $\langle x \rhd y \oplus y, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, with $p\,\mathcal{R}\,p$.

4. $\langle x \rhd y \oplus y, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, with reverse reasoning from the previous case.

A.19 The axiom: $(x \oplus y) \rhd z \approx x \rhd z \oplus y \rhd z$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y, z \in P$:

$$(x \oplus y) \rhd z\,\mathcal{R}\,x \rhd z \oplus y \rhd z \ \wedge\ x\,\mathcal{R}\,x\ .$$

For $x\,\mathcal{R}\,x$, the proof is trivial. For $(x \oplus y) \rhd z\,\mathcal{R}\,x \rhd z \oplus y \rhd z$, we find the following cases.

1. $\langle (x \oplus y) \rhd z, \nu \rangle \checkmark$, for which we need the hypothesis
   (a) $\langle x \oplus y, \nu \rangle \checkmark$, for which we need one of the hypotheses
      i. $\langle x, \nu \rangle \checkmark$
      From which we directly conclude $\langle x \rhd z, \nu \rangle \checkmark$ hence $\langle x \rhd z \oplus y \rhd z, \nu \rangle \checkmark$.
      ii. $\langle y, \nu \rangle \checkmark$
      Similar to the previous case.
2. $\langle x \rhd z \oplus y \rhd z, \nu \rangle \checkmark$, with reverse reasoning from the previous case.
3. $\langle (x \oplus y) \rhd z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need the hypotheses
   (a) $\exists_r\ p = r \blacktriangleright z \wedge \langle x \oplus y, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$, for which we need one of the hypotheses
      i. $\langle x, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$
      From which we conclude $\langle x \rhd z, \nu \rangle \xrightarrow{l} \langle r \blacktriangleright z, \nu' \rangle$ hence $\langle x \rhd z \oplus y \rhd z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, with $p\,\mathcal{R}\,p$.
      ii. $\langle y, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$
      Similar to the previous case.
4. $\langle x \rhd z \oplus y \rhd z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, with reverse reasoning from the previous case.

A.20 The axiom: $(x \rhd y) \rhd z \approx x \rhd (y \blacktriangleright z)$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y, z \in P$:

$$(x \rhd y) \rhd z\,\mathcal{R}\,x \rhd (y \blacktriangleright z) \ \wedge\ (x \blacktriangleright y) \blacktriangleright z\,\mathcal{R}\,x \blacktriangleright (y \blacktriangleright z) \ \wedge\ x\,\mathcal{R}\,x\ .$$

For $x\,\mathcal{R}\,x$, the proof is trivial. For $(x \rhd y) \rhd z\,\mathcal{R}\,x \rhd (y \blacktriangleright z)$, we find the following cases.

1. $\langle (x \rhd y) \rhd z, \nu \rangle \checkmark$, for which we need the hypothesis
   (a) $\langle x \rhd y, \nu \rangle \checkmark$, for which we need the hypothesis
      i. $\langle x, \nu \rangle \checkmark$
         From which we directly conclude $\langle x \rhd (y \blacktriangleright z), \nu \rangle \checkmark$.
2. $\langle x \rhd (y \blacktriangleright z), \nu \rangle \checkmark$, with reverse reasoning from the previous case.
3. $\langle (x \rhd y) \rhd z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need the hypothesis
   (a) $\exists_r\ p = r \blacktriangleright z \wedge \langle x \rhd y, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$, for which we need the hypothesis
      i. $\exists_s\ r = s \blacktriangleright y \wedge \langle x, \nu \rangle \xrightarrow{l} \langle s, \nu' \rangle$
         From which we conclude $\langle x \rhd (y \blacktriangleright z), \nu \rangle \xrightarrow{l} \langle s \blacktriangleright (y \blacktriangleright z), \nu' \rangle$ and $(s \blacktriangleright y) \blacktriangleright z \,\mathcal{R}\, s \blacktriangleright (y \blacktriangleright z)$.
4. $\langle x \rhd (y \blacktriangleright z), \nu \rangle \xrightarrow{l} \langle p', \nu' \rangle$, similar to the previous case.

For $(x \blacktriangleright y) \blacktriangleright z \,\mathcal{R}\, x \blacktriangleright (y \blacktriangleright z)$, we find the following cases.

1. $\langle (x \blacktriangleright y) \blacktriangleright z, \nu \rangle \checkmark$, for which we need one of the hypotheses
   (a) $\langle x \blacktriangleright y, \nu \rangle \checkmark$, for which we need one of the hypotheses
      i. $\langle x, \nu \rangle \checkmark$
         From which we conclude directly $\langle x \blacktriangleright (y \blacktriangleright z), \nu \rangle \checkmark$.
      ii. $\langle y, \nu \rangle \checkmark$ From which we conclude $\langle y \blacktriangleright z, \nu \rangle \checkmark$ and hence $\langle x \blacktriangleright (y \blacktriangleright z), \nu \rangle \checkmark$.
   (b) $\langle z, \nu \rangle \checkmark$
      From which we conclude $\langle y \blacktriangleright z, \nu \rangle \checkmark$ and hence $\langle x \blacktriangleright (y \blacktriangleright z), \nu \rangle \checkmark$.
2. $\langle x \blacktriangleright (y \blacktriangleright z), \nu \rangle \checkmark$, similar to the previous case.
3. $\langle (x \blacktriangleright y) \blacktriangleright z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need one of the hypotheses
   (a) $\langle z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$
      From which we conclude $\langle y \blacktriangleright z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ and hence $\langle x \blacktriangleright (y \blacktriangleright z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.
   (b) $\exists_r\ p = r \blacktriangleright z \wedge \langle x \blacktriangleright y, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$, for which we need one of the hypotheses
      i. $\langle y, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$
         From which we conclude $\langle y \blacktriangleright z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ and hence $\langle x \blacktriangleright (y \blacktriangleright z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.

ii. $\exists_s\ r = s \blacktriangleright y \ \wedge\ \langle x, \nu \rangle \xrightarrow{l} \langle s, \nu' \rangle$
From which we conclude $p = (s \blacktriangleright y) \blacktriangleright z$ and $\langle x \blacktriangleright (y \blacktriangleright z), \nu \rangle \xrightarrow{l} \langle s \blacktriangleright (y \blacktriangleright z), \nu' \rangle$ with $(s \blacktriangleright y) \blacktriangleright z \,\mathcal{R}\, s \blacktriangleright (y \blacktriangleright z)$.

4. $\langle x \blacktriangleright (y \blacktriangleright z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, similar to the previous case.

A.21 The axiom: $(x \mathbin{⫟} y) \mathbin{⫟} z \approx x \mathbin{⫟} (y \parallel z)$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y, z \in P$:

$$(x \mathbin{⫟} y) \mathbin{⫟} z \,\mathcal{R}\, x \mathbin{⫟} (y \parallel z) \ \wedge\ (x \parallel y) \parallel z \,\mathcal{R}\, x \parallel (y \parallel z) \ \wedge\ \ x \,\mathcal{R}\, x\ .$$

For $x \,\mathcal{R}\, x$, the proof is trivial. For $(x \mathbin{⫟} y) \mathbin{⫟} z \,\mathcal{R}\, x \mathbin{⫟} (y \parallel z)$, we find the following cases.

1. $\langle (x \mathbin{⫟} y) \mathbin{⫟} z, \nu \rangle \checkmark$, which cannot be derived.

2. $\langle x \mathbin{⫟} (y \parallel z), \nu \rangle \checkmark$, which cannot be derived.

3. $\langle (x \mathbin{⫟} y) \mathbin{⫟} z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need the hypothesis that $l \in \mathbb{A}$ and furthermore

   (a) $\exists_r\ p = r \parallel z \wedge \langle x \mathbin{⫟} y, \nu \rangle \overset{l}{\mapsto} \langle r, \nu' \rangle$, for which we need the hypothesis

      i. $\exists_s\ r = s \parallel y \ \wedge\ \langle x, \nu \rangle \overset{l}{\mapsto} \langle s, \nu' \rangle$
      From which we conclude $\langle x \mathbin{⫟} (y \parallel z), \nu \rangle \xrightarrow{l} \langle s \parallel (y \parallel z), \nu' \rangle$, with $p = (s \parallel y) \parallel z \,\mathcal{R}\, s \parallel (y \parallel z)$.

4. $\langle x \mathbin{⫟} (y \parallel z), \nu \rangle \xrightarrow{l} \langle p', \nu' \rangle$, similar to the previous case.

For $(x \parallel y) \parallel z \,\mathcal{R}\, x \parallel (y \parallel z)$, we find the following cases.

1. $\langle (x \parallel y) \parallel z, \nu \rangle \checkmark$, for which we need the hypothesis.

   (a) $\langle x \parallel y, \nu \rangle \checkmark \ \wedge\ \langle z, \nu \rangle \checkmark$, for which we need the hypothesis

      i. $\langle x, \nu \rangle \checkmark \ \wedge\ \langle y, \nu \rangle \checkmark$
      From which we conclude directly $\langle x \parallel (y \parallel z), \nu \rangle \checkmark$.

2. $\langle x \parallel (y \parallel z), \nu \rangle \checkmark$, similar to the previous case.

3. $\langle (x \parallel y) \parallel z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need one of the hypotheses

   (a) $\exists_r\ p = (x \parallel y) \parallel r \ \wedge\ l \in \mathbb{A} \ \wedge\ \langle z, \nu \rangle \overset{l}{\mapsto} \langle r, \nu' \rangle$
   From which we conclude $\langle x \parallel (y \parallel z), \nu \rangle \xrightarrow{l} \langle x \parallel (y \parallel r), \nu' \rangle$ with $(x \parallel y) \parallel r \,\mathcal{R}\, x \parallel (y \parallel r)$.

   (b) $\exists_r\ p = r \parallel z \ \wedge\ l \in \mathbb{A} \ \wedge\ \langle (x \parallel y), \nu \rangle \overset{l}{\mapsto} \langle r, \nu' \rangle$, for which we find one of the hypotheses

i. $\exists_s\ r = s \,\|\, y \ \wedge\ \langle x, \nu \rangle \overset{l}{\mapsto} \langle s, \nu' \rangle$ From which we conclude $\langle x \,\|\, (y \,\|\, z), \nu \rangle \xrightarrow{l} \langle s \,\|\, (y \,\|\, z), \nu' \rangle$ with $(s \,\|\, y) \,\|\, z \,\mathcal{R}\, s \,\|\, (y \,\|\, z)$.

ii. $\exists_s\ r = x \,\|\, s \ \wedge\ \langle y, \nu \rangle \overset{l}{\mapsto} \langle s, \nu' \rangle$ From which we conclude $\langle x \,\|\, (y \,\|\, z), \nu \rangle \xrightarrow{l} \langle x \,\|\, (s \,\|\, z), \nu' \rangle$ with $(x \,\|\, s) \,\|\, z \,\mathcal{R}\, x \,\|\, (s \,\|\, z)$.

iii. $\exists_{s,s',a,a',v}\ r = s \,\|\, s' \ \wedge\ l = (a\gamma a', v') \ \wedge\ \langle x, \nu \rangle \overset{a,v}{\mapsto} \langle s, \nu' \rangle \ \wedge$ $\langle y, \nu \rangle \overset{a',v}{\mapsto} \langle s', \nu' \rangle$. From which we conclude $\langle x \,\|\, (y \,\|\, z), \nu \rangle \xrightarrow{l}$ $\langle s \,\|\, (s' \,\|\, z), \nu' \rangle$ with $(s \,\|\, s') \,\|\, z \,\mathcal{R}\, s \,\|\, (s' \,\|\, z)$.

(c) $\exists_{r,r',a,a',v}\ p = r \,\|\, r' \ \wedge\ l = (a\gamma a', v) \ \wedge\ \langle (x \,\|\, y), \nu \rangle \overset{a,v'}{\mapsto} \langle r, \nu' \rangle \ \wedge$ $\langle z, \nu \rangle \overset{a',v}{\mapsto} \langle r', \nu' \rangle$, for which we find one of the hypotheses

i. $\exists_s\ r = s \,\|\, y \ \wedge\ \langle x, \nu \rangle \overset{a,v}{\mapsto} \langle s, \nu' \rangle$ From which we conclude $\langle x \,\|\, (y \,\|\, z), \nu \rangle \xrightarrow{l} \langle s \,\|\, (y \,\|\, r'), \nu' \rangle$ with $(s \,\|\, y) \,\|\, r' \,\mathcal{R}\, s \,\|\, (y \,\|\, r')$.

ii. $\exists_s\ r = x \,\|\, s \ \wedge\ \langle y, \nu \rangle \overset{a,v}{\mapsto} \langle s, \nu' \rangle$ From which we conclude $\langle x \,\|\, (y \,\|\, z), \nu \rangle \xrightarrow{l} \langle x \,\|\, (s \,\|\, r'), \nu' \rangle$ with $(x \,\|\, s) \,\|\, r' \,\mathcal{R}\, x \,\|\, (s \,\|\, r')$.

iii. $\exists_{s,s',a'',a'''}\ r = s \,\|\, s' \ \wedge\ a = a''\gamma a''' \ \wedge\ \langle x, \nu \rangle \overset{a'',v}{\mapsto} \langle s, \nu' \rangle \ \wedge$ $\langle y, \nu \rangle \overset{a''',v}{\mapsto} \langle s', \nu \rangle$. Using associativity of γ, we conclude that $(a''\gamma a''')\gamma a' = a''\gamma(a'''\gamma a')$, from which we conclude in turn that $\langle x \,\|\, (y \,\|\, z), \nu \rangle \xrightarrow{l} \langle s \,\|\, (s' \,\|\, r'), \nu' \rangle$ with $(s \,\|\, s') \,\|\, r' \,\mathcal{R}\, s \,\|\, (s' \,\|\, r')$.

(d) $\langle x \,\|\, y, \nu \rangle \checkmark \ \wedge\ l \in \Sigma \ \wedge\ \langle z, \nu \rangle \overset{l}{\rightsquigarrow} \langle p, \nu' \rangle$, for which we need the hypothesis

- $\langle x, \nu \rangle \checkmark \ \wedge\ \langle y, \nu \rangle \checkmark$

  From which we directly conclude $\langle x \,\|\, (y \,\|\, z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.

(e) $\langle z, \nu \rangle \checkmark \ \wedge\ l \in \Sigma \ \wedge\ \langle x \,\|\, y, \nu \rangle \overset{l}{\rightsquigarrow} \langle p, \nu' \rangle$, for which we need one of the hypotheses

- $\langle x, \nu \rangle \checkmark \ \wedge\ \langle y, \nu \rangle \overset{l}{\rightsquigarrow} \langle p, \nu' \rangle$

  From which we immediately conclude $\langle x \,\|\, (y \,\|\, z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.

- $\langle y, \nu \rangle \checkmark \ \wedge\ \langle x, \nu \rangle \overset{l}{\rightsquigarrow} \langle p, \nu' \rangle$

  From which we immediately conclude $\langle x \,\|\, (y \,\|\, z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.

- $\exists_{r,r'}\ p = r \,\|\, r' \ \wedge\ \langle x, \nu \rangle \overset{l}{\rightsquigarrow} \langle r, \nu' \rangle \ \wedge\ \langle y, \nu \rangle \overset{l}{\rightsquigarrow} \langle r', \nu' \rangle$

  From which we immediately conclude $\langle x \,\|\, (y \,\|\, z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.

(f) $\exists_{r,s}\ p = r \,\|\, s \ \wedge\ l \in \Sigma \ \wedge\ \langle (x \,\|\, y), \nu \rangle \overset{l}{\rightsquigarrow} \langle r, \nu' \rangle \ \wedge\ \langle z, \nu \rangle \overset{l}{\rightsquigarrow} \langle s, \nu' \rangle$, for which we need one of the hypotheses

- $\langle x, \nu \rangle \surd \ \wedge \ \langle y, \nu \rangle \overset{l}{\rightsquigarrow} \langle r, \nu' \rangle$
  From which we immediately conclude $\langle x \,\|\, (y \,\|\, z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.
- $\langle y, \nu \rangle \surd \ \wedge \ \langle x, \nu \rangle \overset{l}{\rightsquigarrow} \langle r, \nu' \rangle$
  From which we immediately conclude $\langle x \,\|\, (y \,\|\, z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.
- $\exists_{r', r''}\ r = r' \,\|\, r'' \ \wedge \ \langle x, \nu \rangle \overset{l}{\rightsquigarrow} \langle r', \nu' \rangle \ \wedge \ \langle y, \nu \rangle \overset{l}{\rightsquigarrow} \langle r'', \nu' \rangle$
  From which we immediately conclude $\langle x \,\|\, (y \,\|\, z), \nu \rangle \xrightarrow{l} \langle r' \,\|\, (r'' \,\|\, s), \nu' \rangle$ with $p = (r' \,\|\, r'') \,\|\, s \,\mathcal{R}\, r' \,\|\, (r'' \,\|\, s)$.

4. $\langle x \,\|\, (y \,\|\, z), \nu \rangle \xrightarrow{l} \langle p', \nu' \rangle$, similar to the previous case.

A.22 The axiom: $(x \mid y) \mid z \approx x \mid (y \mid z)$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y, z \in P$:

$$(x \mid y) \mid z \,\mathcal{R}\, x \mid (y \mid z) \ \wedge \ (x \,\|\, y) \,\|\, z \,\mathcal{R}\, x \,\|\, (y \,\|\, z) \ \wedge \ x \,\mathcal{R}\, x \ .$$

For $x \,\mathcal{R}\, x$, the proof is trivial. For $(x \mid y) \mid z \,\mathcal{R}\, x \mid (y \mid z)$, we find the following cases.

1. $\langle (x \mid y) \mid z, \nu \rangle \surd$, for which we need the hypothesis
   - $\langle x \mid y, \nu \rangle \surd \ \wedge \ \langle z, \nu \rangle \surd$, for which we need the hypothesis
     - $\langle x, \nu \rangle \surd \ \wedge \ \langle y, \nu \rangle \surd$
       From which we immediately conclude $\langle x \mid (y \mid z), \nu \rangle \surd$.
2. $\langle x \mid (y \mid z), \nu \rangle \surd$, similar to the previous case.
3. $\langle (x \mid y) \mid z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need one of the hypotheses

   (a) $\exists_{r, s, a, a', \upsilon}\ p = r \,\|\, s \wedge l = (a \gamma a', \upsilon) \wedge \langle x \mid y, \nu \rangle \overset{a, \upsilon}{\mapsto} \langle r, \nu' \rangle \wedge \langle z, \nu \rangle \overset{a', \upsilon}{\mapsto} \langle s, \nu' \rangle$, for which we need the hypothesis

   i. $\exists_{r', r'', a'', a'''}\ r = r' \,\|\, r'' \ \wedge \ a = a'' \gamma a''' \ \wedge \ \langle x, \nu \rangle \overset{a'', \upsilon}{\mapsto} \langle r', \nu' \rangle \ \wedge \ \langle y, \nu \rangle \overset{a''', \upsilon}{\mapsto} \langle r'', \nu' \rangle$
   Using associativity of γ we then conclude $l = a'' \gamma (a''' \gamma a')$ and hence $\langle x \mid (y \mid z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.

   (b) $\exists_{r, s}\ p = r \,\|\, s \ \wedge \ l \in \Sigma \ \wedge \ \langle x \mid y, \nu \rangle \overset{l}{\rightsquigarrow} \langle r, \nu' \rangle \ \wedge \ \langle z, \nu \rangle \overset{l}{\rightsquigarrow} \langle s, \nu' \rangle$, for which we need one of the hypotheses

   i. $\exists_{r', r''}\ r = r' \,\|\, r'' \ \wedge \ \langle x, \nu \rangle \overset{l}{\rightsquigarrow} \langle r', \nu' \rangle \ \wedge \ \langle y, \nu \rangle \overset{l}{\rightsquigarrow} \langle r'', \nu' \rangle$
   From which we immediately conclude $\langle x \mid (y \mid z), \nu \rangle \xrightarrow{l} \langle r' \,\|\, (r'' \,\|\, s), \nu' \rangle$ with $p = (r' \,\|\, r'') \,\|\, s \,\mathcal{R}\, r' \,\|\, (r'' \,\|\, s)$.

ii. $\langle x, \nu \rangle \overset{l}{\rightsquigarrow} \langle r, \nu' \rangle \ \wedge \ \langle y, \nu \rangle \checkmark$
From which we immediately conclude $\langle x \mid (y \mid z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \mathcal{R} p$.

iii. $\langle y, \nu \rangle \overset{l}{\rightsquigarrow} \langle r, \nu' \rangle \ \wedge \ \langle x, \nu \rangle \checkmark$, similar to the previous case.

(c) $l \in \Sigma \ \wedge \ \langle x \mid y, \nu \rangle \overset{l}{\rightsquigarrow} \langle p, \nu' \rangle \ \wedge \ \langle z, \nu \rangle \checkmark$, for which we need one of the hypotheses

i. $\exists_{r,r'} \ p = r \parallel r' \ \wedge \ \langle x, \nu \rangle \overset{l}{\rightsquigarrow} \langle r, \nu' \rangle \ \wedge \ \langle y, \nu \rangle \overset{l}{\rightsquigarrow} \langle r', \nu' \rangle$
From which we immediately conclude $\langle x \mid (y \mid z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \mathcal{R} p$.

ii. $\langle x, \nu \rangle \overset{l}{\rightsquigarrow} \langle p, \nu' \rangle \ \wedge \ \langle y, \nu \rangle \checkmark$
From which we immediately conclude $\langle x \mid (y \mid z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \mathcal{R} p$.

iii. $\langle y, \nu \rangle \overset{l}{\rightsquigarrow} \langle r, \nu' \rangle \ \wedge \ \langle x, \nu \rangle \checkmark$, similar to the previous case.

(d) $l \in \Sigma \ \wedge \ \langle z, \nu \rangle \overset{l}{\rightsquigarrow} \langle p, \nu' \rangle \ \wedge \ \langle x \mid y, \nu \rangle \checkmark$, for which we need the hypothesis

i. $\langle x, \nu \rangle \checkmark \ \wedge \ \langle y, \nu \rangle \checkmark$
From which we immediately conclude $\langle x \mid (y \mid z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \mathcal{R} p$.

4. $\langle x \mid (y \mid z), \nu \rangle \xrightarrow{l} \langle p', \nu' \rangle$, similar to the previous case.

For $(x \parallel y) \parallel z \mathcal{R} x \parallel (y \parallel z)$, the proof is similar to that of the axiom $(x \mathbin{\|\!\_} y) \mathbin{\|\!\_} z \approx x \mathbin{\|\!\_} (y \parallel z)$.

A.23 The axiom: $(x \mid y) \mathbin{\|\!\_} z \approx x \mid (y \mathbin{\|\!\_} z)$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y, z \in P$:

$$(x \mid y) \mathbin{\|\!\_} z \mathcal{R} x \mid (y \mathbin{\|\!\_} z) \ \wedge \ (x \parallel y) \parallel z \mathcal{R} x \parallel (y \parallel z) \ \wedge \ x \mathcal{R} x \ .$$

For $x \mathcal{R} x$, the proof is trivial.
For $(x \mid y) \mathbin{\|\!\_} z \mathcal{R} x \mid (y \mathbin{\|\!\_} z)$, we find the following cases.

1. $\langle (x \mid y) \mathbin{\|\!\_} z, \nu \rangle \checkmark$, which cannot be satisfied.

2. $\langle x \mid (y \mathbin{\|\!\_} z), \nu \rangle \checkmark$, for which we need the hypothesis

   - $\langle y \mathbin{\|\!\_} z, \nu \rangle \checkmark$, which cannot be satisfied.

3. $\langle (x \mid y) \mathbin{\|\!\_} z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need the hypothesis that $l \in \mathbb{A}$ and furthermore

(a) $\exists_r \ p = r \parallel z \wedge \langle x \mid y, \nu \rangle \overset{l}{\mapsto} \langle r, \nu' \rangle$, for which we need the hypothesis

i. $\exists_{s,s',a,a',\upsilon}\ r = s \,\|\, s' \ \wedge\ l = (a\gamma a', \upsilon) \ \wedge\ \langle x, \nu \rangle \overset{a,\upsilon}{\mapsto} \langle s, \nu' \rangle \ \wedge\ \langle y, \nu \rangle \overset{a',\upsilon}{\mapsto} \langle s', \nu' \rangle$
From which we conclude $\langle x \,|\, (y \,\lfloor\!\lfloor\, z), \nu \rangle \xrightarrow{l} \langle s \,\|\, (s' \,\|\, z), \nu' \rangle$ with $p = (s \,\|\, s') \,\|\, z \,\mathcal{R}\, s \,\|\, (s' \,\|\, z)$.

4. $\langle x \,|\, (y \,\lfloor\!\lfloor\, z), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need one of the hypotheses

   (a) $\exists_{r,s,a,a',\upsilon}\ p = r \,\|\, s \ \wedge\ l = (a\gamma a', \upsilon) \ \wedge\ \langle x, \nu \rangle \overset{a',\upsilon}{\mapsto} \langle r, \nu' \rangle \ \wedge\ \langle y \,\lfloor\!\lfloor\, z, \nu \rangle \overset{a,\upsilon}{\mapsto} \langle s, \nu' \rangle$, for which we need the hypothesis

   i. $\exists_{s'}\ s = s' \,\|\, z \ \wedge\ \langle y, \nu \rangle \overset{a,\upsilon}{\mapsto} \langle r', \nu' \rangle$. From which we conclude $\langle (x \,|\, y) \,\lfloor\!\lfloor\, z, \nu \rangle \xrightarrow{l} \langle (r \,\|\, s') \,\|\, z, \nu' \rangle$ with $(r \,\|\, s') \,\|\, z \,\mathcal{R}\, r \,\|\, (s' \,\|\, z) = p$.

   (b) $\exists_s\ l \in \Sigma \ \wedge\ \langle y \,\lfloor\!\lfloor\, z, \nu \rangle \overset{l}{\leadsto} \langle s, \nu' \rangle$, which cannot be satisfied.

   (c) $l \in \Sigma \ \wedge\ \langle y \,\lfloor\!\lfloor\, z, \nu \rangle \checkmark$, which cannot be satisfied.

For $(x \,\|\, y) \,\|\, z \,\mathcal{R}\, x \,\|\, (y \,\|\, z)$, the proof is similar to that of the axiom $(x \,\lfloor\!\lfloor\, y) \,\lfloor\!\lfloor\, z \approx x \,\lfloor\!\lfloor\, (y \,\|\, z)$.

A.24 The axiom: $(a \odot x) \rhd y \approx a \odot (x \blacktriangleright y)$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y \in P$:

$$(a \odot x) \rhd y \,\mathcal{R}\, a \odot (x \blacktriangleright y) \ \wedge\ (\epsilon \odot x) \blacktriangleright y \,\mathcal{R}\, \epsilon \odot (x \blacktriangleright y) \ \wedge\ x \,\mathcal{R}\, x\ .$$

The proof for $x \,\mathcal{R}\, x$, is trivial. Furthermore, since actions cannot terminate, we find the following cases for $(a \odot x) \rhd y \,\mathcal{R}\, a \odot (x \blacktriangleright y)$.

1. $\langle (a \odot x) \rhd y, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need the hypothesis

   (a) $\exists_r\ p = r \blacktriangleright y \ \wedge\ \langle a \odot x, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$, for which we need the hypothesis

   i. $\exists_s\ r = s \odot x \ \wedge\ \langle a, \nu \rangle \xrightarrow{l} \langle s, \nu' \rangle$, for which we need the hypothesis

   A. $s = \epsilon$
   From which we conclude $p = (\epsilon \odot x) \blacktriangleright y$ and $\langle a \odot (x \blacktriangleright y), \nu \rangle \xrightarrow{l} \langle \epsilon \odot (x \blacktriangleright y), \nu' \rangle$ with $(\epsilon \odot x) \blacktriangleright y \,\mathcal{R}\, \epsilon \odot (x \blacktriangleright y)$.

2. $\langle a \odot (x \blacktriangleright y), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, similar to the previous case.

For $(\epsilon \odot x) \blacktriangleright y \,\mathcal{R}\, \epsilon \odot (x \blacktriangleright y)$, the proof is similar to that of the axiom $\epsilon \odot x \approx x$.

A.25 The axiom: $(d \gg c \rhd x) \odot y \approx d \gg c \rhd x \odot y$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y \in P$:

$(d \gg c \rhd x) \odot y \mathcal{R} d \gg c \rhd x \odot y \;\wedge\; (c \blacktriangleright x) \odot y \mathcal{R} c \blacktriangleright x \odot y \;\wedge\; x \mathcal{R} x$.

The proof for $x \mathcal{R} x$, is trivial. We find the following cases for $(d \gg c \rhd x) \odot y \mathcal{R} d \gg c \rhd x \odot y$.

1. $\langle (d \gg (c \rhd x)) \odot y, \nu \rangle \checkmark$, for which we need the hypothesis
   (a) $\langle d \gg (c \rhd x), \nu \rangle \checkmark \;\wedge\; \langle y, \nu \rangle \checkmark$, for which we need the hypothesis
      i. $\exists_{\nu'} (\nu, \nu') \models d \wedge \langle c \rhd x, \nu' \rangle \checkmark$, for which w need the hypothesis
         A. $\langle c, \nu' \rangle \checkmark$
         Which cannot be derived using the semantical rules of HyPA.
2. $d \gg c \rhd (x \odot y)$ turns out not to terminate, similarly to the previous case.
3. $\langle (d \gg c \rhd x) \odot y, \nu \rangle \xrightarrow{l} \langle p, \nu'' \rangle$, for which we need the hypothesis
   (a) $\exists_r\ p = r \odot y \;\wedge\; \langle d \gg c \rhd x, \nu \rangle \xrightarrow{l} \langle r, \nu'' \rangle$, for which we need the hypothesis
      i. $\exists_{\nu'}\ (\nu, \nu') \models d \wedge \langle c \rhd x, \nu' \rangle \xrightarrow{l} \langle r, \nu'' \rangle$, for which we need the hypothesis
         A. $\exists_s\ r = s \blacktriangleright x \;\wedge\; \langle c, \nu' \rangle \xrightarrow{l} \langle s, \nu'' \rangle$, for which we need the hypothesis
            - $s = c$
              From which we conclude $p = (c \blacktriangleright x) \odot y$ and $\langle c \blacktriangleright x \odot y, \nu' \rangle \xrightarrow{l} \langle c \blacktriangleright x \odot y, \nu'' \rangle$ with $(c \blacktriangleright x) \odot y \mathcal{R} c \blacktriangleright x \odot y$.
   (b) $\langle d \gg c \rhd x \rangle \checkmark \;\wedge\; \langle y, \nu \rangle \xrightarrow{l} \langle p, \nu'' \rangle$, which cannot occur since $d \gg c \rhd x$ does not terminate.
4. $\langle d \gg c \blacktriangleright x \odot y, \nu \rangle \xrightarrow{l} \langle p', \nu'' \rangle$, similar to the previous case.

We find the following cases for $(c \blacktriangleright x) \odot y \mathcal{R} c \blacktriangleright x \odot y$.

1. $\langle (c \blacktriangleright x) \odot y, \nu \rangle \checkmark$, for which we need the hypothesis
   (a) $\langle c \blacktriangleright x, \nu \rangle \checkmark \;\wedge\; \langle y, \nu \rangle \checkmark$, for which we need one of the hypotheses
      i. $\langle c, \nu \rangle \checkmark$
      Which cannot be derived.
      ii. $\langle x, \nu \rangle \checkmark$
      From which we conclude $\langle x \odot y, \nu \rangle \checkmark$ and hence $\langle c \blacktriangleright x \odot y, \nu \rangle \checkmark$.

2. $\langle c \blacktriangleright x \odot y, \nu \rangle \checkmark$, similar to the previous case.

3. $\langle (c \blacktriangleright x) \odot y, \nu \rangle \xrightarrow{l} \langle p, \nu \rangle$, for which we need one of the hypotheses

   (a) $\langle c \blacktriangleright x, \nu \rangle \checkmark \ \wedge \ \langle y, \nu \rangle \xrightarrow{l} \langle p, \nu \rangle$, for which we need one of the hypotheses

      i. $\langle c, \nu \rangle \checkmark$
      Which cannot be derived.

      ii. $\langle x, \nu \rangle \checkmark$
      From which we conclude $\langle x \odot y, \nu \rangle \xrightarrow{l} \langle p, \nu \rangle$ hence $\langle c \blacktriangleright x \odot y, \nu \rangle \xrightarrow{l} \langle p, \nu \rangle$ with $p \mathcal{R} p$.

   (b) $\exists_r \ p = r \odot y \ \wedge \ \langle c \blacktriangleright x, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$, for which we one of the hypotheses

      i. $\langle x, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$
      From which we conclude $\langle x \odot y, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ hence $\langle c \blacktriangleright x \odot y, \nu \rangle \xrightarrow{l} \langle p, \nu \rangle$ with $p \mathcal{R} p$.

      ii. $\exists_s \ r = s \blacktriangleright x \ \wedge \ \langle c, \nu \rangle \xrightarrow{l} \langle s, \nu' \rangle$
      From which we conclude $p = (s \blacktriangleright x) \odot y$ and $\langle c \blacktriangleright x \odot y, \nu \rangle \xrightarrow{l} \langle s \blacktriangleright x \odot y, \nu' \rangle$ with $(s \blacktriangleright x) \odot y \mathcal{R} s \blacktriangleright x \odot y$.

4. $\langle c \blacktriangleright x \odot y, \nu \rangle \xrightarrow{l} \langle p', \nu' \rangle$, similar to the previous case.

A.26 The axiom: $x \rhd \delta \approx x$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x \in P$:

$$x \rhd \delta \mathcal{R} x \ \wedge \ x \blacktriangleright \delta \mathcal{R} x \ .$$

There are only two non-trivial cases:

1. $\langle x \rhd \delta, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need the hypothesis

   (a) $\exists_{p'} \ p = p' \blacktriangleright \delta \ \wedge \ \langle x, \nu \rangle \xrightarrow{l} \langle p', \nu' \rangle$
   For which we conclude $p' \blacktriangleright \delta \mathcal{R} p'$.

2. $\langle x \blacktriangleright \delta, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need the hypothesis

   (a) $\exists_{p'} \ p = p' \blacktriangleright \delta \ \wedge \ \langle x, \nu \rangle \xrightarrow{l} \langle p', \nu' \rangle$
   For which we conclude $p' \blacktriangleright \delta \mathcal{R} p'$.

A.27 The axiom: $\epsilon \rhd x \approx \epsilon$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x \in P$:

$$\epsilon \rhd x \mathcal{R} \epsilon \ .$$

It is trivial to see that $\langle \epsilon \rhd x, \nu \rangle \checkmark \Leftrightarrow \langle \epsilon, \nu \rangle \checkmark$, and that both terms do not generate any transitions. Hence $\mathcal{R}$ is a bisimulation relation.

A.28 The axiom: $(d \gg x) \rhd y \approx d \gg x \rhd y$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x \in P$:

$$(d \gg x) \rhd y \, \mathcal{R} \, d \gg x \rhd y \ \wedge \ x \, \mathcal{R} \, x \ .$$

For $x \, \mathcal{R} \, x$, the proof is trivial. For $(d \gg x) \rhd y \, \mathcal{R} \, d \gg x \rhd y$, we find the following cases.

1. $\langle (d \gg x) \rhd y, \nu \rangle \checkmark$, for which we need the hypothesis
   (a) $\langle d \gg x, \nu \rangle \checkmark$, for which we need the hypothesis
      i. $\exists_{\nu'} \ (\nu, \nu') \models d \ \wedge \ \langle x, \nu' \rangle \checkmark$
         From which we conclude $\langle x \rhd y, \nu' \rangle \checkmark$ hence $\langle d \gg x \rhd y, \nu \rangle \checkmark$.
2. $\langle d \gg x \rhd y, \nu \rangle \checkmark$, similar to the previous case.
3. $\langle (d \gg x) \rhd y, \nu \rangle \xrightarrow{l} \langle p, \nu'' \rangle$, for which we need the hypothesis
   (a) $\exists_r \ p = r \blacktriangleright y \ \wedge \ \langle d \gg x, \nu \rangle \xrightarrow{l} \langle r, \nu'' \rangle$, for which we need the hypothesis
      i. $\exists_{\nu'} \ (\nu, \nu') \models d \ \wedge \ \langle x, \nu' \rangle \xrightarrow{l} \langle r, \nu'' \rangle$
         From which we conclude $\langle x \rhd y, \nu' \rangle \xrightarrow{l} \langle r \blacktriangleright y, \nu'' \rangle$ and hence $\langle d \gg x \rhd y, \nu \rangle \xrightarrow{l} \langle p, \nu'' \rangle$ with $p \, \mathcal{R} \, p$.
4. $\langle d \gg x \rhd y, \nu \rangle \xrightarrow{l} \langle p, \nu'' \rangle$, similar to the previous case.

A.29 The axiom: $x \parallel y \approx x \mathbin{⌊⌊} y \oplus y \mathbin{⌊⌊} x \oplus x \mid y$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y \in P$:

$$x \parallel y \, \mathcal{R} \, x \mathbin{⌊⌊} y \oplus y \mathbin{⌊⌊} x \oplus x \mid y \ \wedge \ x \, \mathcal{R} \, x \ .$$

For $x \, \mathcal{R} \, x$, the proof is trivial. For $x \parallel y \, \mathcal{R} \, x \mathbin{⌊⌊} y \oplus y \mathbin{⌊⌊} x \oplus x \mid y$, we find the following cases.

1. $\langle x \parallel y, \nu \rangle \checkmark$ which leads to:
   (a) $\langle x, \nu \rangle \checkmark \ \wedge \ \langle y, \nu \rangle \checkmark$
       From this we find $\langle x \mid y, \nu \rangle \checkmark$ and hence $\langle x \mathbin{⌊⌊} y \oplus y \mathbin{⌊⌊} x \oplus x \mid y, \nu \rangle \checkmark$.
2. $\langle x \mathbin{⌊⌊} y \oplus y \mathbin{⌊⌊} x \oplus x \mid y, \nu \rangle \checkmark$ which leads to the cases:
   (a) $\langle x \mathbin{⌊⌊} y, \nu \rangle \checkmark$
       For which there is no deduction rule.
   (b) $\langle y \mathbin{⌊⌊} x, \nu \rangle \checkmark$
       For which there is no deduction rule.

(c) $\langle\, x \,|\, y, \nu \,\rangle \checkmark$ which leads to:

i. $\langle\, x, \nu \,\rangle \checkmark \;\wedge\; \langle\, y, \nu \,\rangle \checkmark$
From this we find, directly, $\langle\, x \,\|\, y, \nu \,\rangle \checkmark$.

3. $\langle\, x \,\|\, y, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu' \,\rangle$ which leads to the cases:

(a) $l = (a, v) \;\wedge\; \langle\, x \,\|\, y, \nu \,\rangle \overset{a,v}{\mapsto} \langle\, p, \nu' \,\rangle$ which leads to the cases:

i. $\exists_{p'}\; \langle\, x, \nu \,\rangle \overset{a,v}{\mapsto} \langle\, p', \nu' \,\rangle \;\wedge\; p = p' \,\|\, y$
From which we conclude that $\langle\, x \mathbin{\lfloor\!\lfloor} y, \nu \,\rangle \overset{a,v}{\mapsto} \langle\, p, \nu' \,\rangle$, and hence $\langle\, x \mathbin{\lfloor\!\lfloor} y, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu' \,\rangle$ and $\langle\, x \mathbin{\lfloor\!\lfloor} y \,\oplus\, y \mathbin{\lfloor\!\lfloor} x \,\oplus\, x \,|\, y, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu' \,\rangle$ with $p \,\mathcal{R}\, p$.

ii. $\exists_{p'}\; \langle\, y, \nu \,\rangle \overset{a,v}{\mapsto} \langle\, p', \nu' \,\rangle \;\wedge\; p = x \,\|\, p'$
Similar to the previous case.

iii. $\exists_{p',p'',a',a''}\; p = p' \,\|\, p'' \;\wedge\; a = a'\gamma a'' \;\wedge\; \langle\, x, \nu \,\rangle \overset{a',v}{\mapsto} \langle\, p', \nu' \,\rangle \;\wedge\; \langle\, y, \nu \,\rangle \overset{a'',v}{\mapsto} \langle\, p'', \nu' \,\rangle$
From which we conclude that $\langle\, x \,|\, y, \nu \,\rangle \overset{a'\gamma a'',v}{\mapsto} \langle\, p' \,\|\, p'', \nu' \,\rangle$, and hence $\langle\, x \,|\, y, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu' \,\rangle$ and $\langle\, x \mathbin{\lfloor\!\lfloor} y \,\oplus\, y \mathbin{\lfloor\!\lfloor} x \,\oplus\, x \,|\, y, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu' \,\rangle$ with $p \,\mathcal{R}\, p$.

(b) $l = \sigma \;\wedge\; \langle\, x \,\|\, y, \nu \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p, \nu' \,\rangle$ which leads to the cases:

i. $\langle\, x, \nu \,\rangle \checkmark \;\wedge\; \langle\, y, \nu \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p, \nu' \,\rangle$
From which we conclude that $\langle\, x \,|\, y, \nu \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p, \nu' \,\rangle$ and hence $\langle\, x \,|\, y, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu' \,\rangle$ and $\langle\, x \mathbin{\lfloor\!\lfloor} y \,\oplus\, y \mathbin{\lfloor\!\lfloor} x \,\oplus\, x \,|\, y, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu' \,\rangle$ with $p \,\mathcal{R}\, p$.

ii. $\langle\, y, \nu \,\rangle \checkmark \;\wedge\; \langle\, x, \nu \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p, \nu' \,\rangle$
Similar to the previous case.

iii. $\exists_{p',p''}\; p = p', p'' \,\| \;\wedge\; \langle\, x, \nu \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', \nu' \,\rangle \;\wedge\; \langle\, y, \nu \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'', \nu' \,\rangle$
From which we conclude that $\langle\, x \,|\, y, \nu \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p' \,\|\, p'', \nu' \,\rangle$ and hence $\langle\, x \,|\, y, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu' \,\rangle$ and $\langle\, x \mathbin{\lfloor\!\lfloor} y \,\oplus\, y \mathbin{\lfloor\!\lfloor} x \,\oplus\, x \,|\, y, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu' \,\rangle$ with $p \,\mathcal{R}\, p$.

4. $\langle\, x \mathbin{\lfloor\!\lfloor} y \,\oplus\, y \mathbin{\lfloor\!\lfloor} x \,\oplus\, x \,|\, y, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu \,\rangle$ which leads to the cases:

(a) $\langle\, x \mathbin{\lfloor\!\lfloor} y, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu \,\rangle$
Trivial, since every deduction rule for $\lfloor\!\lfloor$ is also a rule for $\|$.

(b) $\langle\, y \mathbin{\lfloor\!\lfloor} x, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu \,\rangle$
Trivial, since every deduction rule for $\lfloor\!\lfloor$ is also a rule for $\|$.

(c) $\langle\, x \,|\, y, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu \,\rangle$
Trivial, since every deduction rule for $|$ is also a rule for $\|$.

A.30 The axiom: $(x \oplus y) \lfloor\!\lfloor z \approx x \lfloor\!\lfloor z \oplus y \lfloor\!\lfloor z$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y, z \in P$:

$$(x \oplus y) \lfloor\!\lfloor z \mathcal{R} x \lfloor\!\lfloor z \oplus y \lfloor\!\lfloor z \ \wedge \ x \mathcal{R} x \ .$$

For $x \mathcal{R} x$, the proof is trivial. For $(x \oplus y) \lfloor\!\lfloor z \mathcal{R} x \lfloor\!\lfloor z \oplus y \lfloor\!\lfloor z$, we find the following cases.

1. $\langle (x \oplus y) \lfloor\!\lfloor z, \nu \rangle \checkmark$, which cannot occur.
2. $\langle x \lfloor\!\lfloor z \oplus y \lfloor\!\lfloor z, \nu \rangle \checkmark$, for which we need one of the hypotheses:
   (a) $\langle x \lfloor\!\lfloor z, \nu \rangle \checkmark$, which cannot occur.
   (b) $\langle y \lfloor\!\lfloor z, \nu \rangle \checkmark$, which cannot occur.
3. $\langle (x \oplus y) \lfloor\!\lfloor z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need the hypothesis:
   (a) $\exists_r \ p = r \parallel z \ \wedge \ \langle x \oplus y, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$, for which we need the hypothesis:
      i. $\langle x, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$
      From which we may conclude $\langle x \lfloor\!\lfloor z, \nu \rangle \xrightarrow{l} \langle x \parallel z, \nu' \rangle$, hence $\langle x \lfloor\!\lfloor z \oplus y \lfloor\!\lfloor z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \mathcal{R} p$.
      ii. $\langle y, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$
      Which is similar to the previous case.
4. $\langle x \lfloor\!\lfloor z \oplus y \lfloor\!\lfloor z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need one of the hypotheses:
   (a) $\langle x \lfloor\!\lfloor z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need the hypothesis:
      i. $\exists_r \ p = r \parallel z \ \langle x, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$
      From which we readily conclude $\langle x \oplus y, \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$ and hence $\langle (x \oplus y) \lfloor\!\lfloor z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \mathcal{R} p$.
   (b) $\langle y \lfloor\!\lfloor z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$
   Which is similar to the previous case.

A.31 The axiom: $d \gg x \lfloor\!\lfloor y \approx d \gg (x \lfloor\!\lfloor y)$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y \in P$:

$$d \gg x \lfloor\!\lfloor y \mathcal{R} d \gg (x \lfloor\!\lfloor y) \ \wedge \ x \mathcal{R} x \ .$$

For $x \mathcal{R} x$, the proof is trivial. For $d \gg x \lfloor\!\lfloor y \mathcal{R} d \gg (x \lfloor\!\lfloor y)$, we find the following cases.

1. $\langle d \gg x \lfloor\!\lfloor y, \nu \rangle \checkmark$, which cannot occur.

2. $\langle\, d \gg (x \,⌊⌊\, y), \nu \,\rangle \checkmark$, for which we need the hypothesis:

   (a) $\exists_{\nu'}\ (\nu, \nu') \models d \ \wedge\ \langle\, x \,⌊⌊\, y, \nu' \,\rangle \checkmark$, which cannot occur.

3. $\langle\, d \gg x \,⌊⌊\, y, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$, for which we need the hypothesis:

   (a) $\exists_r\ p = r \,\|\, y \ \wedge\ \langle\, d \gg x, \nu \,\rangle \xrightarrow{l} \langle\, r, \nu'' \,\rangle$, for which we need the hypothesis:

      i. $\exists_{\nu'}\ (\nu, \nu') \models d \ \wedge\ \langle\, x, \nu' \,\rangle \xrightarrow{l} \langle\, r, \nu'' \,\rangle$
      From which we may conclude $\langle\, x \,⌊⌊\, y, \nu' \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$ and finally $\langle\, d \gg (x \,⌊⌊\, y), \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$ with $p \,\mathcal{R}\, p$.

4. $\langle\, d \gg (x \,⌊⌊\, y), \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$, for which we need the hypothesis:

   (a) $\exists_{\nu'}\ (\nu, \nu') \models d \ \wedge\ \langle\, x \,⌊⌊\, y, \nu' \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$, for which we need the hypothesis:

      i. $\exists_r\ p = r \,\|\, y \ \wedge\ \langle\, x, \nu' \,\rangle \xrightarrow{l} \langle\, r, \nu'' \,\rangle$
      From which we may conclude $\langle\, d \gg x, \nu \,\rangle \xrightarrow{l} \langle\, r, \nu'' \,\rangle$ and hence $\langle\, d \gg x \,⌊⌊\, y, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu'' \,\rangle$ with $p \,\mathcal{R}\, p$.

A.32 The axiom: $(a \odot x) \,⌊⌊\, y \approx a \odot (x \,\|\, y)$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y \in P$:

$$(a \odot x) \,⌊⌊\, y \,\mathcal{R}\, a \odot (x \,\|\, y) \ \wedge\ (\epsilon \odot x) \,\|\, y \,\mathcal{R}\, \epsilon \odot (x \,\|\, y) \ \wedge\ x \,\mathcal{R}\, x \,.$$

For $x \,\mathcal{R}\, x$, the proof is trivial. For $(a \odot x) \,⌊⌊\, y \,\mathcal{R}\, a \odot (x \,\|\, y)$, we find the following cases.

1. Termination does not occur.

2. $\langle\, (a \odot x) \,⌊⌊\, y, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu' \,\rangle$, for which we need the hypothesis:

   (a) $\exists_r\ p = r \,\|\, y \ \wedge\ \langle\, a \odot x, \nu \,\rangle \xrightarrow{l} \langle\, r, \nu' \,\rangle$, and since actions do not terminate, we need the hypothesis:

      i. $\exists_s\ r = s \odot x \ \wedge\ \langle\, a, \nu \,\rangle \xrightarrow{l} \langle\, s, \nu' \,\rangle$.
      Clearly, $s = \epsilon$, hence $p = (\epsilon \odot x) \,\|\, y$ and $\langle\, a \odot (x \,\|\, y), \nu \,\rangle \xrightarrow{l} \langle\, \epsilon \odot (x \,\|\, y), \nu' \,\rangle$ with $\epsilon \odot (x \,\|\, y) \,\mathcal{R}\, (\epsilon \odot x) \,\|\, y$.

3. $\langle\, a \odot (x \,\|\, y), \nu \,\rangle \xrightarrow{l} \langle\, p, \nu' \,\rangle$, which is similar to the previous case.

For $(\epsilon \odot x) \,\|\, y \,\mathcal{R}\, \epsilon \odot x \,\|\, y$, we find the following cases.

1. $\langle\, (\epsilon \odot x) \,\|\, y, \nu \,\rangle \checkmark$, for which we need the hypothesis:

   (a) $\langle\, \epsilon \odot x, \nu \,\rangle \checkmark \ \wedge\ \langle\, y, \nu \,\rangle \checkmark$, for which we need the hypothesis:

i. $\langle x, \nu \rangle \checkmark$
From which we readily conclude $\langle x \,\|\, y, \nu \rangle \checkmark$, hence $\langle \epsilon \odot (x \,\|\, y), \nu \rangle \checkmark$.

2. $\langle \epsilon \odot (x \,\|\, y), \nu \rangle \checkmark$, which is similar to the previous case.

3. $\langle (\epsilon \odot x) \,\|\, y, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need one of the hypotheses:

   (a) $\exists_{\sigma, p', p''}\ l = \sigma \ \wedge\ p = p' \,\|\, p'' \ \wedge\ \langle \epsilon \odot x, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p', \nu' \rangle \wedge \langle y, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p'', \nu' \rangle$, for which we need the hypothesis:

      i. $\langle x, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p', \nu' \rangle$
      From which we readily conclude $\langle x \,\|\, y, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p' \,\|\, p'', \nu' \rangle$ and hence $\langle \epsilon \odot (x \,\|\, y), \nu \rangle \xrightarrow{\sigma} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.

   (b) $\exists_{\sigma}\ l = \sigma \ \wedge\ \langle \epsilon \odot x, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p, \nu \rangle \wedge \langle y, \nu \rangle \checkmark$, which is similar to the first case.

   (c) $\exists_{\sigma}\ l = \sigma \ \wedge\ \langle \epsilon \odot x, \nu \rangle \checkmark \ \wedge\ \langle y, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p, \nu \rangle$, which is similar to the first case.

   (d) $\exists_{a, a', p', p''}\ l = a \gamma a' \wedge p = p' \,\|\, p'' \wedge \langle \epsilon \odot x, \nu \rangle \overset{a}{\mapsto} \langle p', \nu \rangle \wedge \langle y, \nu \rangle \overset{a'}{\mapsto} \langle p'', \nu' \rangle$, which is similar to the first case.

   (e) $\exists_{a, p'}\ l = a \ \wedge\ p = p' \,\|\, y \ \wedge\ \langle \epsilon \odot x, \nu \rangle \overset{a}{\mapsto} \langle p', \nu \rangle$, which is similar to the first case.

   (f) $\exists_{a, p'}\ l = a \ \wedge\ p = x \,\|\, p' \ \wedge\ \langle y, \nu \rangle \overset{a}{\mapsto} \langle p', \nu \rangle$, which is similar to the first case.

4. $\langle \epsilon \odot (x \,\|\, y), \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, which uses the reverse reasoning of the previous case.

In fact, this last case could also have been concluded from the axiom $\epsilon \odot x \approx x$.

A.33 The axiom: $(c \rhd x) \,⌊\!⌊\, y \approx \delta$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y \in P$:

$$(c \rhd x) \,⌊\!⌊\, y \,\mathcal{R}\, \delta\ .$$

Clearly, both related processes in $(c \rhd x) \,⌊\!⌊\, y \,\mathcal{R}\, \delta$ cannot terminate. Furthermore, for $\langle (c \rhd x) \,⌊\!⌊\, y, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ we ultimately need the hypothesis that $\exists_{a, r}\ l = a \ \wedge\ p = (r \blacktriangleright x) \,\|\, y \ \wedge\ \langle c, \nu \rangle \overset{a}{\mapsto} \langle r, \nu' \rangle$, which clearly cannot occur. Hence both processes do not perform any transitions.

A.34 The axiom: $x \,|\, y \approx y \,|\, x$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y \in P$:

$$x \,|\, y \,\mathcal{R}\, y \,|\, x \ \wedge\ x \,\|\, y \,\mathcal{R}\, y \,\|\, x \ \wedge\ x \,\mathcal{R}\, x.$$

For $x \,\mathcal{R}\, x$, the proof is trivial. For $x \,|\, y \,\mathcal{R}\, y \,|\, x$ and $x \,\|\, y \,\mathcal{R}\, y \,\|\, x$ the proofs are straightforward by symmetry of the rules for $|$ and $\|$.

A.35 The axiom: $(x \oplus y) \mid z \approx x \mid z \oplus y \mid z$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y \in P$:

$$(x \oplus y) \mid z \,\mathcal{R}\, x \mid z \oplus y \mid z \;\wedge\; x \,\mathcal{R}\, x.$$

For $x \,\mathcal{R}\, x$, the proof is trivial. For $(x \oplus y) \mid z \,\mathcal{R}\, x \mid z \oplus y \mid z$, we find the following cases.

1. $\langle (x \oplus y) \mid z, \nu \rangle \checkmark$, which needs the hypothesis:
   (a) $\langle x \oplus y, \nu \rangle \checkmark \;\wedge\; \langle z, \nu \rangle \checkmark$, which needs one of the hypotheses:
      i. $\langle x, \nu \rangle \checkmark$
      From which we readily conclude $\langle x \mid z, \nu \rangle \checkmark$, and hence $\langle x \mid z \oplus y \mid z, \nu \rangle \checkmark$.
      ii. $\langle y, \nu \rangle \checkmark$, which is similar to the previous case.

2. $\langle x \mid z \oplus y \mid z, \nu \rangle \checkmark$, which follows the reverse reasoning from the previous case.

3. $\langle (x \oplus y) \mid z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, which needs one of the hypotheses:
   (a) $\exists_{\sigma, p', p''}\; l = \sigma \;\wedge\; p = p' \,\|\, p'' \;\wedge\; \langle x \oplus y, \nu \rangle \overset{\sigma}{\leadsto} \langle p', \nu' \rangle \;\wedge\; \langle z, \nu \rangle \overset{\sigma}{\leadsto} \langle p'', \nu' \rangle$, which needs one of the hypotheses:
      i. $\langle x, \nu \rangle \overset{\sigma}{\leadsto} \langle p', \nu' \rangle$,
      From which we may conclude that $\langle x \mid z, \nu \rangle \overset{\sigma}{\leadsto} \langle p' \,\|\, p'', \nu' \rangle$ and hence $\langle x \mid z \oplus y \mid z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.
      ii. $\langle y, \nu \rangle \overset{\sigma}{\leadsto} \langle p', \nu' \rangle$,
      similar to the previous case.
   (b) $\exists_{a, a', p', p'', v}\; l = (a \gamma a', v) \;\wedge\; p = p' \,\|\, p'' \;\wedge\; \langle x \oplus y, \nu \rangle \overset{a,v}{\mapsto} \langle p', \nu' \rangle \;\wedge\; \langle z, \nu \rangle \overset{a',v}{\mapsto} \langle p'', \nu' \rangle$,
   which is similar to the previous case.

4. $\langle x \mid z \oplus y \mid z, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, which follows the reverse reasoning of the previous case.

A.36 The axiom: $\delta \mid x \approx \delta$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x \in P$:

$$\delta \mid x \,\mathcal{R}\, \delta.$$

It is straightforward to verify that both terms $\delta \mid x$ and δ do not terminate, nor can perform any transitions.

A.37 The axiom: $\epsilon \parallel x \approx x$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x \in P$:

$$\epsilon \parallel x \,\mathcal{R}\, x \;\wedge\; x \,\mathcal{R}\, x.$$

For $x \,\mathcal{R}\, x$, the proof is trivial. For $\epsilon \parallel x \,\mathcal{R}\, x$, termination is trivial. Furthermore, we have the following cases.

1. $\langle\, \epsilon \parallel x, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu \,\rangle$, which needs one of the hypotheses:
   (a) $\exists_\sigma\; l = \sigma \;\wedge\; \langle\, x, \nu \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p, \nu' \,\rangle$
   From which we easily conclude $\langle\, x, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu' \,\rangle$ and $p \,\mathcal{R}\, p$.
   (b) $\exists_{a,p'}\; p = \epsilon \parallel p' \;\wedge\; \langle\, x, \nu \,\rangle \overset{a}{\mapsto} \langle\, p', \nu' \,\rangle$
   From which we conclude $\langle\, x, \nu \,\rangle \xrightarrow{l} \langle\, p', \nu' \,\rangle$ and $\epsilon \parallel p' \,\mathcal{R}\, p'$.
2. $\langle\, x, \nu \,\rangle \xrightarrow{l} \langle\, p, \nu \,\rangle$, which follows the reverse reasoning of the previous case.

A.38 The axiom: $\epsilon \mathbin{\|\!\!\lfloor} x \approx \delta$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x \in P$:

$$\epsilon \mathbin{\|\!\!\lfloor} x \,\mathcal{R}\, \delta.$$

Clearly, $\mathbin{\|\!\!\lfloor}$ has no rules for termination, and since ϵ cannot perform any transitions, $\epsilon \mathbin{\|\!\!\lfloor} x$ cannot either. This covers the cases of $\epsilon \mathbin{\|\!\!\lfloor} x \,\mathcal{R}\, \delta$.

A.39 The axiom: $d \gg \epsilon \mid d' \gg \epsilon \approx (d^? \wedge d'^?) \gg \epsilon$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that:

$$d \gg \epsilon \mid d' \gg \epsilon \,\mathcal{R}\, (d^? \wedge d'^?) \gg \epsilon.$$

Clearly, we only need to verify the cases of $d \gg \epsilon \mid d' \gg \epsilon \,\mathcal{R}\, (d^? \wedge d'^?) \gg \epsilon$ for termination.

1. $\langle\, d \gg \epsilon \mid d' \gg \epsilon, \nu \,\rangle \checkmark$, for which we need the hypothesis
   (a) $\exists_{\nu'}\; (\nu, \nu') \models d \;\wedge\; \exists_{\nu''}\; (\nu, \nu'') \models d'$
   From which we conclude that $(\nu, \nu) \models d^?$ and $(\nu, \nu) \models d'^?$, hence $\langle\, (d^? \wedge d'^?) \gg \epsilon, \nu \,\rangle \checkmark$.
2. $\langle\, (d^? \wedge d'^?) \gg \epsilon, \nu \,\rangle \checkmark$, for which we need the hypothesis
   (a) $\exists_{\nu'}\; (\nu, \nu') \models (d^? \wedge d'^?)$, which comes down to the hypothesis
      i. $\nu = \nu' \;\wedge\; \exists_{\upsilon}\; (\nu, \upsilon) \models d \;\wedge\; \exists_{\upsilon'}\; (\nu, \upsilon') \models d'$
      From which we easily conclude $\langle\, d \gg \epsilon, \nu \,\rangle \checkmark$ and $\langle\, d' \gg \epsilon, \nu \,\rangle \checkmark$, hence $\langle\, d \gg \epsilon \mid d' \gg \epsilon, \nu \,\rangle \checkmark$.

A.40 The axiom: $d \gg \epsilon \,|\, d' \gg a \odot x \approx \delta$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x \in P$:

$$d \gg \epsilon \,|\, d' \gg a \odot x \,\mathcal{R}\, \delta.$$

Clearly, $d' \gg a \odot x$ does not terminate directly, hence $d \gg \epsilon \,|\, d' \gg a \odot x$, does not. Also $d \gg \epsilon$ does not execute any transitions, and $d' \gg a \odot x$ does not execute any signal transitions, hence $d \gg \epsilon \,|\, d' \gg a \odot x$ does not execute any transitions.

A.41 The axiom: $d \gg a \odot x \,|\, d' \gg a' \odot y \approx (d \wedge d') \gg (a\gamma a') \odot (x \,\|\, y)$ if $(a\gamma a')$ defined

For a given a and a' such that $a\gamma a'$ is defined, we study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y \in P$:

$$d \gg a \odot x \,|\, d' \gg a' \odot y \,\mathcal{R}\, (d \wedge d') \gg (a\gamma a') \odot (x \,\|\, y) \;\wedge\; \epsilon \odot x \,\|\, \epsilon \odot y \,\mathcal{R}\, \epsilon \odot (x \,\|\, y) \;\wedge\; x \,\mathcal{R}\, x.$$

For $x \,\mathcal{R}\, x$, the proof is trivial. For $d \gg a \odot x \,|\, d' \gg a' \odot y \,\mathcal{R}\, (d \wedge d') \gg (a\gamma a') \odot (x \,\|\, y)$, there is clearly no termination. Hence we find only the following cases.

1. $\langle d \gg a \odot x \,|\, d' \gg a' \odot y, \nu \rangle \xrightarrow{l} \langle p, \nu''' \rangle$, for which we need one of the hypotheses:
   (a) $\exists_{\sigma,p',p''}\; l = \sigma \wedge \langle d \gg a \odot x, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p', \nu''' \rangle \wedge \langle d' \gg a' \odot y, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p'', \nu''' \rangle$,
   for which the hypotheses can clearly not be fulfilled.
   (b) $\exists_{a'',a''',p',p'',v}\; l = (a''\gamma a''', v) \wedge \langle d \gg a \odot x, \nu \rangle \overset{a'',v}{\mapsto} \langle p', \nu''' \rangle \wedge \langle d' \gg a' \odot y, \nu \rangle \overset{a''',v}{\mapsto} \langle p'', \nu''' \rangle$, for which we need the hypothesis:
      i. $\exists_{\nu'}\; (\nu, \nu') \models d \wedge \langle a \odot x, \nu' \rangle \overset{a'',v}{\mapsto} \langle p', \nu''' \rangle \wedge \exists_{\nu''}\; (\nu, \nu'') \models d' \wedge \langle a' \odot x, \nu' \rangle \overset{a''',v}{\mapsto} \langle p'', \nu''' \rangle$, for which we need the hypothesis:
         A. $\exists_r\; p' = r \odot x \wedge \langle a, \nu' \rangle \overset{a'',v}{\mapsto} \langle r, \nu''' \rangle \wedge \exists_{r'}\; p'' = r' \odot x \wedge \langle a', \nu' \rangle \overset{a''',v}{\mapsto} \langle r', \nu''' \rangle$, which can only be concluded if:
            - $a = a''$, $a' = a'''$, $v = \nu' = \nu'' = \nu'''$, $r = r' = \epsilon$,
            from which we conclude that $l = (a\gamma a', \nu''')$, $p = \epsilon \odot x \,\|\, \epsilon \odot y$ and $(\nu, \nu''') \models (d \wedge d')$. Finally, we then obtain $\langle (d \wedge d') \gg (a\gamma a') \odot (x \,\|\, y), \nu \rangle \xrightarrow{l} \langle \epsilon \odot (x \,\|\, y), \nu''' \rangle$ and $\epsilon \odot x \,\|\, \epsilon \odot y \,\mathcal{R}\, \epsilon \odot (x \,\|\, y)$.

2. $\langle (d \wedge d') \gg (a\gamma a') \odot (x \,\|\, y), \nu \rangle \xrightarrow{l} \langle p, \nu'' \rangle$, for which we need the hypothesis
   (a) $\exists_{\nu'}\, (\nu, \nu') \models (d \wedge d') \wedge \langle (a\gamma a') \odot (x \,\|\, y), \nu' \rangle \xrightarrow{l} \langle p, \nu'' \rangle$, for which we need the hypothesis:

i. $\exists_r\ p = r \odot (x \,\|\, y) \ \wedge\ \langle (a\gamma a'), \nu' \rangle \xrightarrow{l} \langle r, \nu'' \rangle$, which can only be concluded if:

A. $l = (a\gamma a', \upsilon) \ \wedge\ \upsilon = \nu' = \nu'' \ \wedge\ r = \epsilon$, hence $(\nu, \nu'') \models d$ and $(\nu, \nu'') \models d'$, thus $\langle d \gg a \odot x \,|\, d' \gg a' \odot y, \nu \rangle \xrightarrow{l} \langle \epsilon \odot x \,\|\, \epsilon \odot y, \nu'' \rangle$ and $\epsilon \odot x \,\|\, \epsilon \odot y \,\mathcal{R}\, \epsilon \odot (x \,\|\, y)$.

For $\epsilon \odot x \,\|\, \epsilon \odot y \,\mathcal{R}\, \epsilon \odot (x \,\|\, y)$, the cases are similar to the cases of $\epsilon \odot x \,\|\, y \,\mathcal{R}\, \epsilon \odot (x \,\|\, y)$ in the proof of $(a \odot x) \,\|\!\!\_\, y \approx a \odot (x \,\|\, y)$.

A.42 The axiom: $d \gg a \odot x \,|\, d' \gg a' \odot y \approx \delta$ if $(a\gamma a')$ undefined

For a given a and a' such that $a\gamma a'$ is undefined, we study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y \in P$:

$$d \gg a \odot x \,|\, d' \gg a' \odot y \,\mathcal{R}\, \delta.$$

For $d \gg a \odot x \,|\, d' \gg a' \odot y$ there is clearly no termination. Furthermore, there are no signal transitions. Hence the only case to be studied is: $\langle d \gg a \odot x \,|\, d' \gg a' \odot y, \nu \rangle \overset{a''}{\mapsto} \langle p, \nu' \rangle$, for which we ultimately need the hypothesis that $a'' = a\gamma a'$, which is contradictory to the assumption that $a\gamma a'$ is undefined.

A.43 The axiom: $d \gg \epsilon \,|\, d' \gg c \rhd x \approx (d^? \sim d') \gg c \rhd x$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x \in P$:

$$d \gg \epsilon \,|\, d' \gg c \rhd x \,\mathcal{R}\, (d^? \sim d') \gg c \rhd x \ \wedge\ x \,\mathcal{R}\, x.$$

For $x \,\mathcal{R}\, x$, the proof is trivial. Also, since $c \rhd x$ cannot terminate, the cases for termination are also trivial (both terms cannot). Hence, we only study the following cases for $d \gg \epsilon \,|\, d' \gg c \rhd x \,\mathcal{R}\, (d^? \sim d') \gg c \rhd x$.

1. $\langle d \gg \epsilon \,|\, d' \gg c \rhd x, \nu \rangle \xrightarrow{l} \langle p, \nu'' \rangle$, for which we need one of the hypotheses

   (a) $\exists_{p'}\ l \in \mathbb{A} \wedge\ p = d \gg \epsilon \,\|\, p' \wedge\ \langle d' \gg c \rhd x, \nu \rangle \overset{l}{\mapsto} \langle p', \nu'' \rangle$, which would need

   i. $\exists_{\nu'} (\nu, \nu') \models d \ \wedge\ \langle c \blacktriangleright x, \nu' \rangle \overset{l}{\mapsto} \langle p, \nu'' \rangle$, and hence would need

   A. $\exists_r\ p' = r \blacktriangleright x \ \wedge\ \langle c, \nu' \rangle \overset{l}{\mapsto} \langle r, \nu'' \rangle$, which can clearly not be satisfied.

   (b) $\exists_\sigma\ l = \sigma \ \wedge\ \langle d \gg \epsilon, \nu \rangle \checkmark \ \wedge\ \langle d' \gg c \rhd x, \nu \rangle \overset{\sigma}{\leadsto} \langle p, \nu'' \rangle$, for which we need the hypothesis

   i. $\exists_\upsilon\ (\nu, \upsilon) \models d \ \wedge\ \exists_{\nu'}\ (\nu, \nu') \models d' \ \wedge\ \langle c \rhd x, \nu' \rangle \overset{\sigma}{\leadsto} \langle p, \nu'' \rangle$, but then we may conclude $(\nu, \nu) \models d^?$ hence $(\nu, \nu') \models d^? \sim d')$ and finally $\langle (d^? \wedge d') \gg c \rhd x, \nu \rangle \xrightarrow{\sigma} \langle p, \nu'' \rangle$ with $p \,\mathcal{R}\, p$.

2. $\langle (d^? \wedge d') \gg c \vartriangleright x, \nu \rangle \xrightarrow{l} \langle p, \nu''' \rangle$, for which we need the hypothesis

   (a) $\exists_{\nu''}\ (\nu, \nu'') \models (d^? \sim d')\ \wedge\ \langle c \vartriangleright x, \nu'' \rangle \xrightarrow{l} \langle p, \nu''' \rangle$, which leads to the hypothesis

      i. $\exists_{\nu'} (\nu, \nu') \models d^?\ \wedge\ (\nu', \nu'') \models d'$, for which we need

         A. $\exists_{\upsilon}\ (\nu, \upsilon) \models d\ \wedge \nu = \nu'$.
         Finally, we may conclude that $\langle d \gg \epsilon, \nu \rangle \checkmark$ and $\langle d' \gg c \vartriangleright x, \nu \rangle \xrightarrow{l} \langle p, \nu''' \rangle$, hence $\langle d \gg \epsilon \,|\, d' \gg c \vartriangleright x, \nu \rangle \xrightarrow{l} \langle p, \nu''' \rangle$ with $p \mathcal{R} p$.

A.44 The axiom: $d \gg c \vartriangleright x \,|\, d' \gg a \odot y \approx \delta$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y \in P$:

$$d \gg c \vartriangleright x \,|\, d' \gg a \odot y \approx \delta.$$

Clearly, $c \vartriangleright x$ cannot terminate, nor perform any action transitions, while $a \odot y$ cannot terminate and cannot perform any signal transitions. None of the hypothesis of the forced-synchronization operator can therefore be fulfilled.

A.45 The axiom: $d \gg c \vartriangleright x \,|\, d' \gg c' \vartriangleright y \approx ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \vartriangleright (x \,\|\!\!\_\, c' \blacktriangleright y \oplus y \,\|\!\!\_\, c \blacktriangleright x \oplus x \,|\, c' \blacktriangleright y \oplus y \,|\, c \blacktriangleright x)$

In this section, we will use the abbreviations $M = \begin{pmatrix} x \,\|\!\!\_\, c' \blacktriangleright y \oplus \\ y \,\|\!\!\_\, c \blacktriangleright x \oplus \\ x \,|\, c' \blacktriangleright y \oplus \\ y \,|\, c \blacktriangleright x \end{pmatrix}$ and $N = ((d \sim c_{jmp})\ \wedge (d' \sim c'_{jmp}))$.

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y \in P$:

- $d \gg c \vartriangleright x \,|\, d' \gg c' \vartriangleright y \mathcal{R} N \gg c \wedge c' \vartriangleright M$
- $c \blacktriangleright x \,\|\, c' \blacktriangleright y \mathcal{R} c \wedge c' \blacktriangleright M$
- $x \,\|\, y \mathcal{R} y \,\|\, x$
- $x \mathcal{R} x$

For $x \mathcal{R} x$, the proof is trivial.

For $x \,\|\, y \mathcal{R} y \,\|\, x$, the proof follows according to the same lines as in the proof of axiom $x \,|\, y \mathcal{R} y \,|\, x$.

For $d \gg c \vartriangleright x \,|\, d' \gg c' \vartriangleright y \mathcal{R} N \gg c \wedge c' \vartriangleright M$, we find the following cases.

1. $\langle d \gg c \vartriangleright x \,|\, d' \gg c' \vartriangleright y, \nu \rangle \checkmark$, for which we need the hypothesis

(a) $\langle d \gg c \rhd x, \nu \rangle \checkmark \;\wedge\; \langle d' \gg c' \rhd y, \nu \rangle \checkmark$, which leads to the hypothesis

i. $\exists_{\nu'}\, (\nu, \nu') \models d \wedge \langle c \rhd x, \nu' \rangle \checkmark$, for which we need the hypothesis

A. $\langle c, \nu' \rangle \checkmark$, which cannot be satisfied.

2. $\langle N \gg c \wedge c' \rhd M, \nu \rangle \checkmark$, cannot be satisfied for similar reasons as in the previous case.

3. $\langle d \gg c \rhd x \,|\, d' \gg c' \rhd y, \nu \rangle \xrightarrow{l} \langle p, \nu''' \rangle$, leading to one of the hypotheses

(a) $\exists_{p'}\; l \in \mathbb{A} \;\wedge\; \langle d \gg c \rhd x, \nu \rangle \overset{l}{\mapsto} \langle p', \nu''' \rangle$, which can clearly not be satisfied since flow-clauses cannot execute action transitions.

(b) $\exists_{\sigma, p', p''}\; l = \sigma \;\wedge\; dom(\sigma) = [0 \ldots t] \;\wedge\; p = p' \,\|\, p'' \;\wedge\; \langle d \gg c \rhd x, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p', \nu''' \rangle \;\wedge\; \langle d' \gg c' \rhd y, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p'', \nu''' \rangle$, for which we need the hypothesis

i. $\exists_{\nu'} (\nu, \nu') \models d \;\wedge\; \langle c \rhd x, \nu' \rangle \overset{\sigma}{\rightsquigarrow} \langle p', \nu''' \rangle \;\wedge\; \exists_{\nu''}\, (\nu, \nu'') \models d' \;\wedge\; \langle c' \rhd y, \nu'' \rangle \overset{\sigma}{\rightsquigarrow} \langle p'', \nu''' \rangle$, leading to the hypothesis

A. $\exists_{r'}\; p' = r' \blacktriangleright x \;\wedge\; \langle c, \nu' \rangle \overset{\sigma}{\rightsquigarrow} \langle r', \nu''' \rangle \;\wedge\; \exists_{r''}\; p'' = r'' \blacktriangleright y \;\wedge\; \langle c, \nu'' \rangle \overset{\sigma}{\rightsquigarrow} \langle r'', \nu''' \rangle$, for which we need the hypothesis

- $(\nu', \sigma) \models c \wedge r' = c \wedge (\nu'', \sigma) \models c' \wedge r'' = c' \wedge \nu''' = \sigma(t)$. Using the second lemma on the solutions of clauses, found in the proof of axiom $c_{jmp} \gg c \approx c$, we know that $(\sigma(0), \sigma) \models (c \wedge c')$. Furthermore, we may conclude that $(\nu, \sigma(0)) \models N$ and $p = c \blacktriangleright x \,\|\, c' \blacktriangleright y$, to finally find $\langle N \gg c \wedge c' \rhd M, \nu \rangle \xrightarrow{l} \langle (c \wedge c') \blacktriangleright M, \nu''' \rangle$ and $p \,\mathcal{R}\, c \wedge c' \blacktriangleright M$.

4. $\langle N \gg (c \wedge c') \rhd M, \nu \rangle \xrightarrow{l} \langle p, \nu'' \rangle$, leading to the hypothesis

(a) $\exists_{\nu'}\, (\nu, \nu') \models N \;\wedge\; \langle (c \wedge c') \rhd M, \nu' \rangle \xrightarrow{l} \langle p, \nu'' \rangle$, for which we need the hypothesis

i. $\exists_r\, p = r \blacktriangleright M \;\wedge\; \langle (c \wedge c'), \nu' \rangle \xrightarrow{l} \langle r, \nu'' \rangle \;\wedge\; \exists_{\nu_1, \sigma_1}\, (\nu, \nu_1) \models d \;\wedge\; (\nu_1, \sigma_1) \models c \;\wedge\; \exists_{\nu_2, \sigma_2}\, (\nu, \nu_2) \models d' \;\wedge\; (\nu_2, \sigma_2) \models c \;\wedge\; \nu' = \sigma_1(0) = \sigma_2(0)$, and finally we need the hypothesis

A. $l = \sigma \;\wedge\; r = (c \wedge c') \;\wedge\; (\nu', \sigma) \models (c \wedge c') \;\wedge\; \nu'' = \sigma(t)$.
From this we may conclude that $p = (c \wedge c') \blacktriangleright M$, but furthermore we can use the first lemma on solutions of flow clauses, found in the proof of axiom $c_{jmp} \gg c \approx c$, together with the facts that $(\nu_1, \sigma_1) \models c$ and $(\nu', \sigma) \models c$ and $\nu' = \sigma_1(0)$ to find $(\nu_1, \sigma) \models c$ and similarly $(\nu_2, \sigma) \models c'$. This leads to the observations that $\langle d \gg c \rhd x \rangle \overset{\sigma}{\rightsquigarrow} \langle c \blacktriangleright x, \nu'' \rangle$ and $\langle d' \gg c' \rhd y \rangle \overset{\sigma}{\rightsquigarrow} \langle c' \blacktriangleright y, \nu'' \rangle$, and finally $\langle d \gg c \rhd x \,|\, d' \gg c' \rhd y, \nu \rangle \xrightarrow{l} \langle c \blacktriangleright x \,\|\, c \blacktriangleright y, \nu'' \rangle$ and $c \blacktriangleright x \,\|\, c \blacktriangleright y \,\mathcal{R}\, p$.

For $c \blacktriangleright x \,\|\, c' \blacktriangleright y \,\mathcal{R}\, c \wedge c' \blacktriangleright M$, we find the following cases.

1. $\langle c \blacktriangleright x \,\|\, c' \blacktriangleright y, \nu \rangle \checkmark$, for which we need the hypothesis
   (a) $\langle c \blacktriangleright x, \nu \rangle \checkmark \;\wedge\; \langle c' \blacktriangleright y, \nu \rangle \checkmark$, for which we need the hypothesis
      i. $\langle x, \nu \rangle \checkmark \;\wedge\; \langle y, \nu \rangle \checkmark$
      From which we may conclude $\langle x \,|\, y, \nu \rangle \checkmark$ hence $\langle M, \nu \rangle \checkmark$ and $\langle (c \wedge c') \blacktriangleright M, \nu \rangle \checkmark$.

2. $\langle c \wedge c' \blacktriangleright M, \nu \rangle \checkmark$, for which we need the hypothesis $\langle M, \nu \rangle \checkmark$ and hence one of the following hypotheses
   (a) $\langle x \,\lfloor\!\lfloor\, c' \blacktriangleright y, \nu \rangle \checkmark$, which cannot occur.
   (b) $\langle y \,\lfloor\!\lfloor\, c \blacktriangleright x, \nu \rangle \checkmark$, which cannot occur.
   (c) $\langle x \,|\, c' \blacktriangleright y, \nu \rangle \checkmark$, for which we need the hypothesis
      i. $\langle x \rangle \checkmark \;\wedge\; \langle c' \blacktriangleright y, \nu \rangle \checkmark$. From this we may conclude that $\langle c \blacktriangleright x, \nu \rangle \checkmark$ and hence $\langle c \blacktriangleright x \,\|\, c' \blacktriangleright y, \nu \rangle \checkmark$.
   (d) $\langle y \,|\, c \blacktriangleright x, \nu \rangle \checkmark$, is similar to the previous case.

3. $\langle c \blacktriangleright x \,\|\, c' \blacktriangleright y, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need one of the following hypotheses:
   (a) $\exists_{a,a',p',p''}\; l = a\gamma a'\; p = p' \,\|\, p'' \;\wedge\; \langle c \blacktriangleright x, \nu \rangle \overset{a}{\mapsto} \langle p', \nu' \rangle \;\wedge\; \langle c' \blacktriangleright y, \nu \rangle \overset{a'}{\mapsto} \langle p'', \nu' \rangle$, which leads to the hypothesis
      i. $\langle x, \nu \rangle \overset{a}{\mapsto} \langle p', \nu' \rangle$,
      from which we conclude $\langle x \,|\, c' \blacktriangleright y, \nu \rangle \overset{a\gamma a'}{\mapsto} \langle p' \,\|\, p'', \nu' \rangle$, and hence $\langle (c \wedge c') \blacktriangleright M, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.
   (b) $\exists_{a,p'}\; l = a\; p = p' \,\|\, c' \blacktriangleright y \wedge \langle c \blacktriangleright x \rangle \overset{a}{\mapsto} \langle p', \nu' \rangle$, for which we need the hypothesis
      i. $\langle x, \nu \rangle \overset{a}{\mapsto} \langle p', \nu' \rangle$
      from which we conclude that $\langle x \,\lfloor\!\lfloor\, c' \blacktriangleright y, \nu \rangle \xrightarrow{a} \langle p' \,\|\, c' \blacktriangleright y, \nu' \rangle$ and hence $\langle (c \wedge c') \blacktriangleright M, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.
   (c) $\exists_{a,p'}\; l = a\; p = c \blacktriangleright x \,\|\, p' \wedge \langle c' \blacktriangleright y \rangle \overset{a}{\mapsto} \langle p', \nu' \rangle$, which is similar to the previous case.
   (d) $\exists_{\sigma,p',p''}\; l = \sigma \;\wedge\; dom(\sigma) = [0 \ldots t] \;\wedge\; p = p' \,\|\, p'' \;\wedge\; \langle c \blacktriangleright x, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p', \nu' \rangle \;\wedge\; \langle c' \blacktriangleright y, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p'', \nu' \rangle$, for which we need one of the following hypotheses:
      i. $\exists_{r'}\; p' = r' \blacktriangleright x \;\wedge\; \langle c, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle r', \nu' \rangle \;\wedge\; \exists_{r''}\; p'' = r'' \blacktriangleright y \;\wedge\; \langle c', \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle r'', \nu' \rangle$, for which we need the hypothesis
         A. $r' = c \;\wedge\; r'' = c' \;\wedge\; (\nu, \sigma) \models c \;\wedge\; (\nu, \sigma) \models c' \;\wedge\; \nu' = \sigma(t)$
         From this we conclude that $p = c \blacktriangleright x \,\|\, c' \blacktriangleright y$ and $\langle (c \wedge c') \blacktriangleright M, \nu \rangle \xrightarrow{l} \langle c \wedge c' \blacktriangleright M, \nu' \rangle$, with $p \,\mathcal{R}\, (c \wedge c') \blacktriangleright M$.

ii. $\exists_{r'}\ p' = r' \blacktriangleright x \ \wedge\ \langle c, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle r', \nu' \rangle \ \wedge\ \langle y, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p'', \nu' \rangle$, for which we need the hypothesis

A. $r' = c$.

Now, we conclude that $p = c \blacktriangleright x \,\|\, p''$, and that $\langle y \,|\, c \blacktriangleright x, \nu \rangle \xrightarrow{l} \langle p'' \,\|\, c \blacktriangleright x, \nu' \rangle$. Hence $\langle (c \wedge c') \blacktriangleright M, \nu \rangle \xrightarrow{l} \langle p'' \,\|\, c \blacktriangleright x, \nu' \rangle$ with $p'' \,\|\, c \blacktriangleright x \,\mathcal{R}\, p$.

iii. $\langle x, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p', \nu' \rangle \ \wedge\ \exists_{r''}\ p'' = r'' \blacktriangleright y \ \wedge\ \langle c', \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle r'', \nu' \rangle$, for which we need the hypothesis

A. $r'' = c'$.

Now, we conclude that $p = p' \,\|\, c' \blacktriangleright y$, and that $\langle x \,|\, c' \blacktriangleright y, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$. Hence, $\langle (c \wedge c') \blacktriangleright M, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.

iv. $\langle x, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p', \nu' \rangle \ \wedge\ \langle y, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p'', \nu' \rangle$

From which it follows directly that $\langle x \,|\, c' \blacktriangleright y \rangle \xrightarrow{l} \langle p, \nu' \rangle$ Hence, $\langle (c \wedge c') \blacktriangleright M, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.

(e) $\exists_{\sigma}\ l = \sigma \ \wedge\ \langle c \blacktriangleright x, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p, \nu' \rangle \ \wedge\ \langle c' \blacktriangleright y, \nu \rangle \checkmark$, for which we need the hypothesis

i. $\langle y, \nu \rangle \checkmark$.

From this we may conclude $\langle y \,|\, c \blacktriangleright x \rangle \xrightarrow{l} \langle p, \nu' \rangle$. Hence, $\langle (c \wedge c') \blacktriangleright M, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.

(f) $\exists_{\sigma}\ l = \sigma \ \wedge\ \langle c \blacktriangleright x, \nu \rangle \checkmark \ \wedge\ \langle c' \blacktriangleright y, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p, \nu' \rangle$, for which we need the hypothesis

i. $\langle x, \nu \rangle \checkmark$.

From this we may conclude $\langle x \,|\, c' \blacktriangleright y \rangle \xrightarrow{l} \langle p, \nu' \rangle$. Hence, $\langle (c \wedge c') \blacktriangleright M, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.

4. $\langle c \wedge c' \blacktriangleright M, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, which needs one of the following hypotheses:

(a) $\exists_{r}\ p = r \blacktriangleright M \ \wedge\ \langle (c \wedge c'), \nu \rangle \xrightarrow{l} \langle r, \nu' \rangle$, for which we need the hypothesis

i. $\exists_{\sigma}\ l = \sigma \wedge dom(\sigma) = [0 \ldots t] \wedge r = (c \wedge c') \wedge \nu' = \sigma(t) \wedge (\nu, \sigma) \models c \ \wedge\ (\nu, \sigma) \models c'$.

From this we may readily conclude that $p = (c \wedge c') \blacktriangleright M$ and $\langle c \blacktriangleright x, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle c \blacktriangleright x, \nu' \rangle$. Consequently, we find $\langle c \blacktriangleright x \,\|\, c' \blacktriangleright y, \nu \rangle \xrightarrow{l} \langle c \blacktriangleright x \,\|\, c' \blacktriangleright y, \nu' \rangle$ with $c \blacktriangleright x \,\|\, c' \blacktriangleright y \,\mathcal{R}\, p$.

(b) $\langle M, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, which comes down to one of the hypotheses:

i. $\langle x \,\|\!\!\_\, c' \blacktriangleright y, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for this we need the hypothesis

A. $\exists_{r,a}\ l = a \ \wedge\ p = r \,\|\, c' \blacktriangleright y \ \wedge\ \langle x, \nu \rangle \overset{a}{\mapsto} \langle r, \nu' \rangle$.

From which we conclude $\langle c \blacktriangleright x \rangle \overset{a}{\mapsto} \langle r, \nu' \rangle$ and finally $\langle c \blacktriangleright x \,\|\, c' \blacktriangleright y, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.

ii. $\langle y \,\|\, c \blacktriangleright x, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for this we need the hypothesis

  A. $\exists_{r,a}\ l = a \ \wedge\ p = r \,\|\, c \blacktriangleright x \ \wedge\ \langle y, \nu \rangle \overset{a}{\mapsto} \langle r, \nu' \rangle$.
  From which we conclude $\langle c \blacktriangleright y \rangle \overset{a}{\mapsto} \langle r, \nu' \rangle$ and finally $\langle c \blacktriangleright x \,\|\, c' \blacktriangleright y, \nu \rangle \xrightarrow{l} \langle c \blacktriangleright x \,\|\, r, \nu' \rangle$ with $p \,\mathcal{R}\, c \blacktriangleright x \,\|\, r$.

iii. $\langle x \,|\, c' \blacktriangleright y, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need one of the hypotheses

  A. $\exists_{a,a',p',p''}\ l = a\gamma a'\ p = p' \,\|\, p'' \ \wedge\ \langle x, \nu \rangle \overset{a}{\mapsto} \langle p', \nu' \rangle \ \wedge\ \langle c' \blacktriangleright y, \nu \rangle \overset{a'}{\mapsto} \langle p'', \nu' \rangle$, which leads to the hypothesis

  - $\langle y, \nu \rangle \overset{a'}{\mapsto} \langle p'', \nu' \rangle$
    From which we readily conclude $\langle c \blacktriangleright x \,\|\, c' \blacktriangleright y, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.

  B. $\exists_{\sigma,p',p''}\ l = \sigma\ p = p' \,\|\, p'' \ \wedge\ \langle x, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p', \nu' \rangle \ \wedge\ \langle c' \blacktriangleright y, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p'', \nu' \rangle$, which leads to one of the hypotheses

  - $\exists_r\ p'' = r \blacktriangleright y \ \wedge\ \langle c' \blacktriangleright y, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle r, \nu' \rangle$, then we need the hypothesis
    - $r = c'$
      From which we conclude that $p = p' \,\|\, c' \blacktriangleright y$ and $\langle c \blacktriangleright x \,\|\, c' \blacktriangleright y, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.
  - $\langle y, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p'', \nu' \rangle$
    From which we readily conclude $\langle c \blacktriangleright x \,\|\, c' \blacktriangleright y, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$ with $p \,\mathcal{R}\, p$.

iv. $\langle y \,|\, c \blacktriangleright x, \nu \rangle \xrightarrow{l} \langle p, \nu' \rangle$, for which we need one of the hypotheses

  A. $\exists_{a,a',p',p''}\ l = a\gamma a'\ p = p' \,\|\, p'' \ \wedge\ \langle y, \nu \rangle \overset{a}{\mapsto} \langle p', \nu' \rangle \ \wedge\ \langle c \blacktriangleright x, \nu \rangle \overset{a'}{\mapsto} \langle p'', \nu' \rangle$, which leads to the hypothesis

  - $\langle x, \nu \rangle \overset{a'}{\mapsto} \langle p'', \nu' \rangle$
    From which we readily conclude $\langle c \blacktriangleright x \,\|\, c' \blacktriangleright y, \nu \rangle \xrightarrow{l} \langle p'' \,\|\, p', \nu' \rangle$ with $p \,\mathcal{R}\, p'' \,\|\, p'$.

  B. $\exists_{\sigma,p',p''}\ l = \sigma\ p = p' \,\|\, p'' \ \wedge\ \langle y, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p', \nu' \rangle \ \wedge\ \langle c' \blacktriangleright x, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p'', \nu' \rangle$, which leads to one of the hypotheses

  - $\exists_r\ p'' = r \blacktriangleright x \ \wedge\ \langle c \blacktriangleright x, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle r, \nu' \rangle$, then we need the hypothesis
    - $r = c$
      From which we conclude that $p = p' \,\|\, c \blacktriangleright x$ and $\langle c \blacktriangleright x \,\|\, c' \blacktriangleright y, \nu \rangle \xrightarrow{l} \langle c \blacktriangleright x \,\|\, p', \nu' \rangle$ with $p \,\mathcal{R}\, c \blacktriangleright x \,\|\, p'$.
  - $\langle x, \nu \rangle \overset{\sigma}{\rightsquigarrow} \langle p'', \nu' \rangle$
    From which we readily conclude $\langle c \blacktriangleright x \,\|\, c' \blacktriangleright y, \nu \rangle \xrightarrow{l} \langle p'' \,\|\, p', \nu' \rangle$ with $p \,\mathcal{R}\, p'' \,\|\, p'$.

A.46 The axiom: $\partial_H(x \oplus y) \approx \partial_H(x) \oplus \partial_H(y)$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y \in P$:

$$\partial_H(x \oplus y)\,\mathcal{R}\,\partial_H(x) \oplus \partial_H(y) \;\wedge\; x\,\mathcal{R}\,x.$$

For $x\,\mathcal{R}\,x$, the proof is trivial. For $\partial_H(x \oplus y)\,\mathcal{R}\,\partial_H(x) \oplus \partial_H(y)$, we have the following cases.

1. $\langle\, \partial_H(x \oplus y), \nu \,\rangle \checkmark$, for which we need the hypothesis
   (a) $\langle\, x \oplus y, \nu \,\rangle \checkmark$, for which we need one of the hypotheses
      i. $\langle\, x, \nu \,\rangle \checkmark$,
      from which we conclude $\langle\, \partial_H(x), \nu \,\rangle \checkmark$, hence $\langle\, \partial_H(x) \oplus \partial_H(y), \nu \,\rangle \checkmark$.
      ii. $\langle\, y, \nu \,\rangle \checkmark$,
      similar to the previous case.
2. $\langle\, \partial_H(x) \oplus \partial_H(y), \nu \,\rangle \checkmark$,
   which is similar to the previous case.
3. $\langle\, \partial_H(x \oplus y), \nu \,\rangle \xrightarrow{l} \langle\, p, \nu' \,\rangle$, for which we need one of the hypotheses
   (a) $\exists_{\sigma,r}\; l = \sigma \;\wedge\; p = \partial_H(r)\;\; \langle\, x \oplus y, \nu \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, r, \nu' \,\rangle$, which leads to one of the hypotheses
      i. $\langle\, x, \nu \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, r, \nu' \,\rangle$
      From which we conclude $\langle\, \partial_H(x), \nu \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p, \nu' \,\rangle$ and hence $\langle\, \partial_H(x) \oplus \partial_H(y), \nu \,\rangle \xrightarrow{l} \langle\, p, \nu' \,\rangle$ with $p\,\mathcal{R}\,p$.
      ii. $\langle\, y, \nu \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p, \nu' \,\rangle$, which is similar to the previous case.
   (b) $\exists_{a,v,r}\; l = (a, v) \;\wedge\; p = \partial_H(r) \;\wedge\; a \notin H \;\wedge\; \langle\, x \oplus y, \nu \,\rangle \overset{l}{\mapsto} \langle\, r, \nu' \,\rangle$,
   for which the proof is similar to the previous case.

A.47 The axiom: $\partial_H(x \odot y) \approx \partial_H(x) \odot \partial_H(y)$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y \in P$:

$$\partial_H(x \odot y)\,\mathcal{R}\,\partial_H(x) \odot \partial_H(y) \;\wedge\; x\,\mathcal{R}\,x.$$

The proof of which follows roughly the same lines as the previous.

A.48 The axiom: $\partial_H(x \rhd y) \approx \partial_H(x) \rhd \partial_H(y)$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x, y \in P$:

$$\partial_H(x \rhd y)\,\mathcal{R}\,\partial_H(x) \rhd \partial_H(y) \;\wedge\; x\,\mathcal{R}\,x.$$

The proof of which follows roughly the same lines as the previous.

A.49 The axiom: $\partial_H(d \gg x) \approx d \gg \partial_H(x)$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that for all $x \in P$:

$$\partial_H(d \gg x)\ \mathcal{R}\ d \gg \partial_H(x) \ \wedge\ x\,\mathcal{R}\,x.$$

The proof of which follows roughly the same lines as the previous.

A.50 The axiom: $\partial_H(a) \approx a$ if $a \notin H$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that, if $a \notin H$, then:

$$\partial_H(a)\ \mathcal{R}\ a \ \wedge\ \partial_H(\epsilon)\ \mathcal{R}\ \epsilon.$$

The proof that $\mathcal{R}$ is a bisimulation relation goes as follows.

Clearly, $\langle \partial_H(\epsilon) \rangle \checkmark$ if and only if $\langle \epsilon \rangle \checkmark$, while both cannot execute any transitions.

Furthermore, $\partial_H(a)$ and a both cannot terminate, and if $a \notin H$ they perform the transitions $\langle \partial_H(a), \nu \rangle \overset{a}{\mapsto} \langle \partial_H(\epsilon), \nu \rangle$ and $\langle a, \nu \rangle \overset{a}{\mapsto} \langle \epsilon, \nu \rangle$, respectively. Clearly $\partial_H(\epsilon)\ \mathcal{R}\ \epsilon$.

A.51 The axiom: $\partial_H(a) \approx \delta$ if $a \in H$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that, if $a \in H$, then:

$$\partial_H(a)\ \mathcal{R}\ \delta.$$

The proof that $\mathcal{R}$ is a bisimulation relation goes as follows.

Clearly, both terms cannot terminate, nor can they execute signal transitions, and the one of the hypotheses needed for $\langle \partial_H(a), \nu \rangle \overset{a}{\mapsto} \langle p, \nu' \rangle$ is that $a \notin H$, which does not hold by assumption.

A.52 The axiom: $\partial_H(\epsilon) \approx \epsilon$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that:

$$\partial_H(\epsilon)\ \mathcal{R}\ \epsilon.$$

It is straightforward to verify that $\mathcal{R}$ is a bisimulation relation.

A.53 The axiom: $\partial_H(c) \approx c$

We study the smallest relation $\mathcal{R} \subseteq P \times P$ such that:

$$\partial_H(c)\ \mathcal{R}\ c.$$

Clearly, both processes cannot terminate. Furthermore, the observation that flow-clauses can only execute signal transitions, makes the rest of the proof straightforward.

B Conservativity of ACP

B.1 ACP $\vdash p \approx q$ implies HyPA $\vdash p \approx q$

The following paragraphs contain, for each axiom of ACP, a derivation in HyPA. Together with the observation that the derivation rules of ACP are contained in those of HyPA we find that ACP $\vdash p \approx q$ implies HyPA $\vdash p \approx q$. Note, that in the axioms of ACP, every action a may be replaced by deadlock δ. In HyPA this is not the case. Therefore, we have two versions for some of the axioms on communication and encapsulation.

The axiom: $x \oplus y \approx y \oplus x$ Trivial.

The axiom: $(x \oplus y) \oplus z \approx x \oplus (y \oplus z)$ Trivial.

The axiom: $x \oplus x \approx x$

$$\begin{array}{rcl} x \oplus x & \approx & [\,\mathit{true}\,] \gg x \oplus x \\ & \approx & [\,\mathit{true}\,] \gg x \oplus [\,\mathit{true}\,] \gg x \\ & \approx & [\,\mathit{true}\,] \vee [\,\mathit{true}\,] \gg x \\ & \approx & [\,\mathit{true}\,] \gg x \\ & \approx & x \end{array}$$

The axiom: $(x \oplus y) \odot z \approx x \odot z \oplus y \odot z$ Trivial.

The axiom: $(x \odot y) \odot z \approx x \odot (y \odot z)$ Trivial.

The axiom: $x \oplus \delta \approx x$

$$\begin{array}{rcl} x \oplus \delta & \approx & x \oplus [\,\mathit{false}\,] \gg x \\ & \approx & [\,\mathit{true}\,] \gg x \oplus [\,\mathit{false}\,] \gg x \\ & \approx & [\,\mathit{true}\,] \vee [\,\mathit{false}\,] \gg x \\ & \approx & [\,\mathit{true}\,] \gg x \\ & \approx & x \end{array}$$

The axiom: $\delta \odot x \approx \delta$

$$\begin{array}{rcl} \delta \odot x & \approx & ([\,\mathit{false}\,] \gg \epsilon) \odot x \\ & \approx & [\,\mathit{false}\,]^{?} \gg x \\ & \approx & [\,\mathit{false}\,] \gg x \\ & \approx & \delta \end{array}$$

The axiom: $a \mid b \approx a\gamma b$, if $a\gamma b$ defined In this proof, we use the derivation of $x \oplus \delta \approx x$.

$$\begin{array}{rcl}
a \mid b & \approx & a \odot \epsilon \mid b \\
& \approx & a \odot \epsilon \mid b \odot \epsilon \\
& \approx & [\,\mathit{true}\,] \gg a \odot \epsilon \mid b \odot \epsilon \\
& \approx & [\,\mathit{true}\,] \gg a \odot \epsilon \mid\ [\,\mathit{true}\,] \gg b \odot \epsilon \\
& \approx & ([\,\mathit{true}\,] \wedge [\,\mathit{true}\,]) \gg (a\gamma b) \odot (\epsilon \parallel \epsilon) \\
& \approx & [\,\mathit{true}\,] \gg (a\gamma b) \odot (\epsilon \parallel \epsilon) \\
& \approx & (a\gamma b) \odot (\epsilon \parallel \epsilon) \\
& \approx & (a\gamma b) \odot \epsilon \\
& \approx & a\gamma b
\end{array}$$

The axiom: $a \mid b \approx \delta$, if $a\gamma b$ undefined

$$\begin{array}{rcl}
a \mid b & \approx & a \odot \epsilon \mid b \\
& \approx & a \odot \epsilon \mid b \odot \epsilon \\
& \approx & [\,\mathit{true}\,] \gg a \odot \epsilon \mid b \odot \epsilon \\
& \approx & [\,\mathit{true}\,] \gg a \odot \epsilon \mid\ [\,\mathit{true}\,] \gg b \odot \epsilon \\
& \approx & \delta
\end{array}$$

If we replace a or b by δ we trivially find $a \mid \delta \approx \delta \mid b \approx \delta$.

The axiom: $x \parallel y \approx x \mathbin{|\!\lfloor} y \oplus y \mathbin{|\!\lfloor} x \oplus x \mid y$ Trivial.

The axiom: $a \mathbin{|\!\lfloor} x \approx a \odot x$

$$\begin{array}{rcl}
a \mathbin{|\!\lfloor} x & \approx & (a \odot \epsilon) \mathbin{|\!\lfloor} x \\
& \approx & a \odot (\epsilon \parallel x) \\
& \approx & a \odot x
\end{array}$$

If we replace a by δ we trivially find $\delta \mathbin{|\!\lfloor} x = \delta$.

The axiom: $a \odot x \mathbin{|\!\lfloor} y \approx a \odot (x \parallel y)$ Trivial. Furthermore, if we replace a by δ, we easily find the following derivation.

$$\begin{array}{rcl}
\delta \odot x \mathbin{|\!\lfloor} y & \approx & \delta \mathbin{|\!\lfloor} y \\
& \approx & \delta \\
& \approx & \delta \odot (x \parallel y)
\end{array}$$

The axiom: $(x \oplus y) \mathbin{⌊\!⌊} z \approx x \mathbin{⌊\!⌊} z \oplus y \mathbin{⌊\!⌊} z$ Trivial.

The axiom: $a \odot x \mid b \approx (a \mid b) \odot x$ The proof of this has four cases. If $a\gamma b$ is defined, we obtain the following proof, in which we use $a \mid b = a\gamma b$.

$$\begin{aligned}
a \odot x \mid b &\approx a \odot x \mid b \odot \epsilon \\
&\approx [\,\mathit{true}\,] \gg a \odot x \mid b \odot \epsilon \\
&\approx [\,\mathit{true}\,] \gg a \odot x \mid [\,\mathit{true}\,] \gg b \odot \epsilon \\
&\approx ([\,\mathit{true}\,] \wedge [\,\mathit{true}\,]) \gg (a\gamma b) \odot (x \parallel \epsilon) \\
&\approx [\,\mathit{true}\,] \gg (a\gamma b) \odot (x \parallel \epsilon) \\
&\approx (a\gamma b) \odot (x \parallel \epsilon) \\
&\approx (a\gamma b) \odot (x \mathbin{⌊\!⌊} \epsilon \oplus \epsilon \mathbin{⌊\!⌊} x \oplus x \mathbin{⌊\!⌊} \epsilon) \\
&\approx (a\gamma b) \odot (\epsilon \mathbin{⌊\!⌊} x \oplus x \mathbin{⌊\!⌊} \epsilon \oplus x \mathbin{⌊\!⌊} \epsilon) \\
&\approx (a\gamma b) \odot (\epsilon \mathbin{⌊\!⌊} x \oplus x \mathbin{⌊\!⌊} \epsilon \oplus \epsilon \mathbin{⌊\!⌊} x) \\
&\approx (a\gamma b) \odot (\epsilon \parallel x) \\
&\approx (a\gamma b) \odot x \\
&\approx (a \mid b) \odot x
\end{aligned}$$

If $a\gamma b$ is undefined, we obtain the following proof, in which we use $a \mid b = \delta$.

$$\begin{aligned}
a \odot x \mid b &\approx a \odot x \mid b \odot \epsilon \\
&\approx [\,\mathit{true}\,] \gg a \odot x \mid b \odot \epsilon \\
&\approx [\,\mathit{true}\,] \gg a \odot x \mid [\,\mathit{true}\,] \gg b \odot \epsilon \\
&\approx \delta \\
&\approx \delta \odot x \\
&\approx (a \mid b) \odot x
\end{aligned}$$

If a is replaced by deadlock, we find

$$\begin{aligned}
\delta \odot x \mid b &\approx \delta \mid b \\
&\approx \delta \\
&\approx \delta \odot x \\
&\approx (\delta \mid b) \odot x
\end{aligned}$$

And similarly if b is replaced by deadlock (using commutativity)..

The axiom: $a \,|\, b \odot x \approx (a \,|\, b) \odot x$ The proof of this has four cases. If $a\gamma b$ is defined, we obtain the following proof, in which we use $a \,|\, b = a\gamma b$.

$$\begin{aligned}
a \,|\, b \odot x &\approx a \odot \epsilon \,|\, b \odot x \\
&\approx a \odot \epsilon \,|\, [\,\mathit{true}\,] \gg b \odot x \\
&\approx [\,\mathit{true}\,] \gg a \odot \epsilon \,|\, [\,\mathit{true}\,] \gg b \odot x \\
&\approx ([\,\mathit{true}\,] \wedge [\,\mathit{true}\,]) \gg (a\gamma b) \odot (\epsilon \,\|\, x) \\
&\approx [\,\mathit{true}\,] \gg (a\gamma b) \odot (\epsilon \,\|\, x) \\
&\approx (a\gamma b) \odot (\epsilon \,\|\, x) \\
&\approx (a\gamma b) \odot x \\
&\approx (a \,|\, b) \odot x
\end{aligned}$$

If $a\gamma b$ is undefined, we obtain the following proof, in which we use $a \,|\, b = \delta$.

$$\begin{aligned}
a \,|\, b \odot x &\approx a \odot \epsilon \,|\, b \odot x \\
&\approx a \odot \epsilon \,|\, [\,\mathit{true}\,] \gg b \odot x \\
&\approx [\,\mathit{true}\,] \gg a \odot \epsilon \,|\, [\,\mathit{true}\,] \gg b \odot x \\
&\approx \delta \\
&\approx \delta \odot x \\
&\approx (a \,|\, b) \odot x
\end{aligned}$$

If a is replaced by deadlock we find

$$\begin{aligned}
\delta \,|\, b \odot x &\approx \delta \\
&\approx \delta \odot x \\
&\approx (a \,|\, b) \odot x
\end{aligned}$$

And similarly if b is replaced by deadlock (using $\delta \odot x \approx \delta$ and commutativity).

The axiom: $a \odot x \,|\, b \odot y \approx (a \,|\, b) \odot (x \,\|\, y)$ The proof of this has four cases. If $a\gamma b$ is defined, we obtain the following proof, in which we use $a \,|\, b = a\gamma b$.

$$\begin{aligned}
a \odot x \,|\, b \odot y &\approx a \odot x \,|\, [\,\mathit{true}\,] \gg b \odot y \\
&\approx [\,\mathit{true}\,] \gg a \odot x \,|\, [\,\mathit{true}\,] \gg b \odot y \\
&\approx ([\,\mathit{true}\,] \wedge [\,\mathit{true}\,]) \gg (a\gamma b) \odot (x \,\|\, y) \\
&\approx [\,\mathit{true}\,] \gg (a\gamma b) \odot (x \,\|\, y) \\
&\approx (a\gamma b) \odot (x \,\|\, y) \\
&\approx (a \,|\, b) \odot (x \,\|\, y)
\end{aligned}$$

If $a\gamma b$ is undefined, we obtain the following proof, in which we use $a \mid b = \delta$.

$$\begin{array}{rcl}
a \odot x \mid b \odot y & \approx & a \odot x \mid \big[\,\mathit{true}\,\big] \gg b \odot y \\
 & \approx & \big[\,\mathit{true}\,\big] \gg a \odot x \mid \big[\,\mathit{true}\,\big] \gg b \odot y \\
 & \approx & \delta \\
 & \approx & \delta \odot (x \,\|\, y) \\
 & \approx & (a \mid b) \odot (x \,\|\, y)
\end{array}$$

If a is replaced by deadlock we find

$$\begin{array}{rcl}
\delta \odot x \mid b \odot y & \approx & \delta \mid b \odot y \\
 & \approx & \delta \\
 & \approx & \delta \odot (x \,\|\, y) \\
 & \approx & (\delta \mid b) \odot (x \,\|\, y)
\end{array}$$

And similarly if b is replaced by deadlock (using commutativity).

The axiom: $(x \oplus y) \mid z \approx x \mid z \oplus y \mid z$ Trivial

The axiom: $x \mid (y \oplus z) \approx x \mid y \oplus x \mid z$

$$\begin{array}{rcl}
x \mid (y \oplus z) & \approx & (y \oplus z) \mid x \\
 & \approx & y \mid x \oplus z \mid x \\
 & \approx & x \mid y \oplus z \mid x \\
 & \approx & x \mid y \oplus x \mid z
\end{array}$$

The axiom: $\partial_H(a) \approx a$, **if** $a \notin H$ Trivial.

The axiom: $\partial_H(a) \approx \delta$, **if** $a \in H$ Trivial, except when a is replaced by deadlock. Then we find the following derivation.

$$\begin{array}{rcl}
\partial_H(\delta) & \approx & \partial_H(\mathit{false} \gg x) \\
 & \approx & \mathit{false} \gg \partial_H(x) \\
 & \approx & \delta
\end{array}$$

The axiom: $\partial_H(x \oplus y) \approx \partial_H(x) \oplus \partial_H(y)$ Trivial.

The axiom: $\partial_H(x \odot y) \approx \partial_H(x) \odot \partial_H(y)$ Trivial.

B.2 HyPA $\vdash p \approx q$ implies ACP $\vdash p \approx q$

In this section, we will prove the converse case. This is done using the semantical model of HyPA and ACP. We show, that if two closed ACP terms p and q are bisimilar in HyPA (which we may assume using soundness of the derivation HyPA $\vdash p \approx q$), then they are bisimilar in ACP. Then, we use completeness of the axiomatization of ACP, to conclude that there must be a derivation in ACP to show this bisimilarity. Throughout this section, we use the notation $x \in$ ACP for 'x is a closed ACP term', and similarly for HyPA.

The operational semantics of ACP, is given by the following rules.

$$\frac{}{\langle a\rangle \xrightarrow{a}_{ACP} \surd} \qquad \frac{\langle x\rangle \xrightarrow{a}_{ACP} \langle x'\rangle}{\begin{array}{c}\langle x \oplus y\rangle \xrightarrow{a}_{ACP} \langle x'\rangle\\ \langle y \oplus x\rangle \xrightarrow{a}_{ACP} \langle x'\rangle\end{array}} \qquad \frac{\langle x\rangle \xrightarrow{a}_{ACP} \surd}{\begin{array}{c}\langle x \oplus y\rangle \xrightarrow{a}_{ACP} \surd\\ \langle y \oplus x\rangle \xrightarrow{a}_{ACP} \surd\end{array}}$$

$$\frac{\langle x\rangle \xrightarrow{a}_{ACP} \langle x'\rangle}{\langle x \odot y\rangle \xrightarrow{a}_{ACP} \langle x' \odot y\rangle} \qquad \frac{\langle x\rangle \xrightarrow{a}_{ACP} \surd}{\langle x \odot y\rangle \xrightarrow{a}_{ACP} \langle y\rangle}$$

$$\frac{\langle x\rangle \xrightarrow{a}_{ACP} \langle x'\rangle}{\begin{array}{c}\langle x \,\|\, y\rangle \xrightarrow{a}_{ACP} \langle x' \,\|\, y\rangle\\ \langle y \,\|\, x\rangle \xrightarrow{a}_{ACP} \langle y \,\|\, x'\rangle\\ \langle x \mathbin{\lfloor\!\lfloor} y\rangle \xrightarrow{a}_{ACP} \langle x' \,\|\, y\rangle\end{array}} \qquad \frac{\langle x\rangle \xrightarrow{a}_{ACP} \surd}{\begin{array}{c}\langle x \,\|\, y\rangle \xrightarrow{a}_{ACP} \langle y\rangle\\ \langle y \,\|\, x\rangle \xrightarrow{a}_{ACP} \langle y\rangle\\ \langle x \mathbin{\lfloor\!\lfloor} y\rangle \xrightarrow{a}_{ACP} \langle y\rangle\end{array}}$$

$$\frac{\langle x\rangle \xrightarrow{a'}_{ACP} \langle x'\rangle, \langle y\rangle \xrightarrow{a''}_{ACP} \langle y'\rangle,\ a = a'\gamma a''}{\begin{array}{c}\langle x \,\|\, y\rangle \xrightarrow{a}_{ACP} \langle x' \,\|\, y'\rangle\\ \langle x \mid y\rangle \xrightarrow{a}_{ACP} \langle x' \,\|\, y'\rangle\end{array}}$$

$$\frac{\langle x\rangle \xrightarrow{a'}_{ACP} \surd, \langle y\rangle \xrightarrow{a''}_{ACP} \langle y'\rangle,\ a = a'\gamma a''}{\begin{array}{c}\langle x \,\|\, y\rangle \xrightarrow{a}_{ACP} \langle y'\rangle\\ \langle y \,\|\, x\rangle \xrightarrow{a}_{ACP} \langle y'\rangle\\ \langle x \mid y\rangle \xrightarrow{a}_{ACP} \langle y'\rangle\end{array}}$$

$$\frac{\langle x\rangle \xrightarrow{a'}_{ACP} \surd, \langle y\rangle \xrightarrow{a''}_{ACP} \surd,\ a = a'\gamma a''}{\begin{array}{c}\langle x \,\|\, y\rangle \xrightarrow{a}_{ACP} \surd\\ \langle x \mid y\rangle \xrightarrow{a}_{ACP} \surd\end{array}}$$

$$\frac{\langle x\rangle \xrightarrow{a}_{ACP} \langle x'\rangle, a \notin H}{\langle \partial_H(x)\rangle \xrightarrow{a}_{ACP} \langle \partial_H(x')\rangle} \qquad \frac{\langle x\rangle \xrightarrow{a}_{ACP} \surd, a \notin H}{\langle \partial_H(x)\rangle \xrightarrow{a}_{ACP} \surd}$$

Note, that the empty process ϵ is not an ACP term. In stead, ACP has a transition predicate denoted as $\langle p\rangle \xrightarrow{a}_{ACP} \surd$. The notion of bisimulation for ACP terms is therefore defined as follows.

Definition 10 (ACP-Bisimulation) *A relation $\mathcal{R} \subseteq P \times P$ on process terms of ACP, is an* ACP-bisimulation *relation if for all $p, q \in P$ such that $p \,\mathcal{R}\, q$, we find*

- $\langle p \rangle \xrightarrow{a}_{ACP} \langle p' \rangle$ *implies there exists q' s.t. $\langle q \rangle \xrightarrow{a}_{ACP} \langle q' \rangle$ and $p' \,\mathcal{R}\, q'$;*
- $\langle q \rangle \xrightarrow{a}_{ACP} \langle q' \rangle$ *implies there exists p' s.t. $\langle p \rangle \xrightarrow{a}_{ACP} \langle p' \rangle$ and $p' \,\mathcal{R}\, q'$;*
- $\langle p \rangle \xrightarrow{a}_{ACP} \surd$ *implies there exists q' s.t. $\langle q \rangle \xrightarrow{a}_{ACP} \surd$;*
- $\langle q \rangle \xrightarrow{a}_{ACP} \surd$ *implies there exists p' s.t. $\langle p \rangle \xrightarrow{a}_{ACP} \surd$.*

Two process terms x and y are ACP-bisimilar, *denoted $x \leftrightarrows_{ACP} y$, if there exists an ACP-bisimulation relation that relates them.*

Now we will prove the following theorem, relating ACP-bisimulation with bisimulation as defined for HyPA.

Theorem 4 *For closed ACP terms p and q we find that if $p \leftrightarrows q$ then $p \leftrightarrows_{ACP} q$.*

Clearly, using soundness of HyPA and completeness of ACP, we can derive from this theorem that

$$\text{HyPA} \vdash p \approx q \Rightarrow p \leftrightarrows q \Rightarrow p \leftrightarrows_{ACP} q \Rightarrow \text{ACP} \vdash p \approx q.$$

The following four lemmas are used to prove this theorem.

Lemma 3
If $x \in ACP$ and $\langle x \rangle \xrightarrow{a}_{ACP} \surd$ then there exists $y' \leftrightarrows \epsilon$ (with $y' \in HyPA$) such that $\langle x, \nu \rangle \xrightarrow{a,\nu} \langle y', \nu \rangle$ for every $\nu \in Val$.

Proof This proof uses induction on the structure of x. Since $x \in$ ACP, we find the following cases.

1. $x = \delta$, which contradicts with the assumption $\langle x \rangle \xrightarrow{a}_{ACP} \surd$.
2. $x = a$. From which we conclude using the semantics of HyPA that $\langle x, \nu \rangle \xrightarrow{a,\nu} \langle \epsilon, \nu \rangle$ for every $\nu \in Val$.
3. $x = x' \oplus x'' \wedge x', x'' \in$ ACP, for which we find the one of the hypotheses, using the semantics of ACP.
   (a) $\langle x' \rangle \xrightarrow{a}_{ACP} \surd$
   With induction, we conclude for x' that there exists $y' \leftrightarrows \epsilon$ such that $\langle x', \nu \rangle \xrightarrow{a,\nu} \langle y', \nu \rangle$ for every $\nu \in Val$, hence also $\langle x, \nu \rangle \xrightarrow{a,\nu} \langle y', \nu \rangle$, using the semantics of HyPA.
   (b) $\langle x'' \rangle \xrightarrow{a}_{ACP} \surd$, similar to the previous case.
4. $x = x' \odot x''$, which contradicts with the assumption $\langle x \rangle \xrightarrow{a}_{ACP} \surd$.

5. $x = x' \,||\, x'' \;\wedge\; x', x'' \in$ ACP, for which we find the following hypothesis, using the semantics of ACP.

   (a) $\exists_{a', a''}\; a = a'\gamma a'' \;\wedge \langle\, x' \,\rangle \overset{a'}{\rightarrow}_{ACP} \surd \;\wedge\; \langle\, x'' \,\rangle \overset{a''}{\rightarrow}_{ACP} \surd$ With induction, we conclude for x' and x'', that there exists $z \leftrightarrows z' \leftrightarrows \epsilon$ such that $\langle\, x', \nu \,\rangle \overset{a', \nu}{\rightarrow} \langle\, z, \nu \,\rangle$ and $\langle\, x'', \nu \,\rangle \overset{a'', \nu}{\rightarrow} \langle\, z', \nu \,\rangle$ for every $\nu \in Val$. Using the semantics of HyPA, we then find that $\langle\, x, \nu \,\rangle \overset{a, \nu}{\rightarrow} \langle\, z \,||\, z', \nu \,\rangle$ and using congruence for the parallel composition, together with the derivable, hence sound, theorem $\epsilon \,||\, \epsilon \approx \epsilon$, we obtain $z \,||\, z' \approx \epsilon$.

6. $x = x' \,\|\!\!\_\, x''$, which contradicts with the assumption $\langle\, x \,\rangle \overset{a}{\rightarrow}_{ACP} \surd$.

7. $x = x' \,|\, x'' \;\wedge\; x', x'' \in$ ACP, similar to the proof of $x = x' \,||\, x''$.

8. $x = \partial_H(x') \wedge x' \in$ ACP, for which we find the following hypothesis, using the semantics of ACP.

   (a) $a \notin H \;\wedge\; \langle\, x' \,\rangle \overset{a}{\rightarrow}_{ACP} \surd$. With induction, we conclude for x', that there exists $y' \leftrightarrows \epsilon$ such that $\langle\, x', \nu \,\rangle \overset{a, \nu}{\rightarrow} \langle\, y', \nu \,\rangle$ for every $\nu \in Val$, hence also $\langle\, \partial_H(x), \nu \,\rangle \overset{a, \nu}{\rightarrow} \langle\, \partial_H(y'), \nu \,\rangle$ and with congruence and the sound axiom $\partial_\epsilon\, (\approx)\, \epsilon$ we find $\partial_H(y') \leftrightarrows \epsilon$.

$\boxtimes$

Lemma 4

If $x \in$ ACP and $\langle\, x \,\rangle \overset{a}{\rightarrow}_{ACP} \langle\, y \,\rangle$ then there exists $y' \leftrightarrows y$ (with $y' \in$ HyPA) such that $\langle\, x, \nu \,\rangle \overset{a, \nu}{\rightarrow} \langle\, y', \nu \,\rangle$ for every $\nu \in Val$.

Proof This proof uses induction on the structure of x. Since $x \in$ ACP, we find the following cases.

1. $x = \delta$, which contradicts with the assumption $\langle\, x \,\rangle \overset{a}{\rightarrow}_{ACP} \langle\, x' \,\rangle$.

2. $x = a$, which contradicts with the assumption $\langle\, x \,\rangle \overset{a}{\rightarrow}_{ACP} \langle\, x' \,\rangle$.

3. $x = x' \oplus x'' \;\wedge\; x', x'' \in$ ACP, for which we find the one of the following hypotheses, using the semantics of ACP.

   (a) $\langle\, x' \,\rangle \overset{a}{\rightarrow}_{ACP} \langle\, y \,\rangle$
   With induction, we conclude for x', that there exists $y' \leftrightarrows y$ such that $\langle\, x', \nu \,\rangle \overset{a, \nu}{\rightarrow} \langle\, y', \nu \,\rangle$ for every $\nu \in Val$, hence also $\langle\, x, \nu \,\rangle \overset{a, \nu}{\rightarrow} \langle\, y', \nu \,\rangle$, using the semantics of HyPA.

   (b) $\langle\, x'' \,\rangle \overset{a}{\rightarrow}_{ACP} \langle\, y \,\rangle$, similar to the previous case.

4. $x = x' \odot x'' \;\wedge\; x', x'' \in$ ACP, for which we find the one of the following hypotheses, using the semantics of ACP.

(a) $\exists_z\ y = z \odot x'' \ \wedge\ \langle x' \rangle \xrightarrow{a}_{ACP} \langle z \rangle$
With induction, we conclude for x', that there exists a $z' \leftrightarrows z$ such that $\langle x', \nu \rangle \overset{a,\nu}{\to} \langle z', \nu \rangle$ for every $\nu \in \mathit{Val}$, hence we find $\langle x, \nu \rangle \overset{a,\nu}{\to} \langle z' \odot x'', \nu \rangle$ using the semantics of HyPA, and $z' \odot x'' \leftrightarrows y$ using congruence of the sequential composition.

(b) $x'' = y \ \wedge\ \langle x' \rangle \xrightarrow{a}_{ACP} \surd$. Using the previous lemma, we may conclude that there exists $z \leftrightarrows \epsilon$ such that $\langle x', \nu \rangle \overset{a,\nu}{\to} \langle z, \nu \rangle$ for all $\nu \in \mathit{Val}$. Then, using the semantics of HyPA we find $\langle x, \nu \rangle \overset{a,\nu}{\to} \langle z \odot x'', \nu \rangle$. Congruence for the sequential composition, and soundness of the axiom $\epsilon \odot x \approx x$ then give use $z \odot x'' \leftrightarrows y$.

5. $x = x' \parallel x'' \ \wedge\ x, x'' \in$ ACP, for which we find one of the following hypotheses, using the semantics of ACP.

   (a) $\exists_z\ y = z \parallel x'' \ \wedge\ \langle x' \rangle \xrightarrow{a}_{ACP} \langle z \rangle$
   With induction, we conclude for x', that there exists $z' \leftrightarrows z$ such that $\langle x', \nu \rangle \overset{a,\nu}{\to} \langle z', \nu \rangle$ for all $\nu \in \mathit{Val}$, and using the semantics of HyPA we conclude $\langle x, \nu \rangle \overset{a,\nu}{\to} \langle z' \parallel x'', \nu \rangle$. Congruence for the parallel composition then gives us $z' \parallel x'' \leftrightarrows y$.

   (b) $\exists_z\ y = x' \parallel z \ \wedge\ \langle x'' \rangle \xrightarrow{a}_{ACP} \langle z \rangle$, similar to the previous case.

   (c) $\exists_{a', a'', z, z'}\ y = z \parallel z' \wedge a = a' \gamma a'' \wedge \langle x' \rangle \xrightarrow{a'}_{ACP} \langle z \rangle \wedge \langle x'' \rangle \xrightarrow{a''}_{ACP} \langle z' \rangle$ With induction to the structure of x' and x'' we find $w \leftrightarrows z$ and $w' \leftrightarrows z'$ such that $\langle x', \nu \rangle \overset{a',\nu}{\to} \langle w, \nu \rangle$ and $\langle x'', \nu \rangle \overset{a'',\nu}{\to} \langle w', \nu \rangle$. Using the semantics of HyPA we then conclude $\langle x, \nu \rangle \overset{a,\nu}{\to} \langle w \parallel w', \nu \rangle$ and congruence for the parallel composition gives $w \parallel w' \leftrightarrows y$.

6. $x = x' \mathbin{\lfloor\!\lfloor} x''$, a subcase of $x = x' \parallel x''$.

7. $x = x' \mid x''$, a subcase of $x = x' \parallel x''$.

8. $x = \partial_H(x') \wedge x' \in$ ACP, for which we find the following hypothesis, using the semantics of ACP.

   (a) $\exists_z\ y = \partial_H(z) \ \wedge\ a \notin H \ \wedge\ \langle x' \rangle \xrightarrow{a}_{ACP} \langle z \rangle$. With induction, we conclude for x', that there exists $z' \leftrightarrows z$ such that $\langle x', \nu \rangle \overset{a,\nu}{\to} \langle z', \nu \rangle$ for every $\nu \in \mathit{Val}$, hence also $\langle \partial_H(x), \nu \rangle \overset{a,\nu}{\to} \langle \partial_H(z'), \nu \rangle$ and with congruence we find $y \leftrightarrows \partial_H(z')$.

⊠

Lemma 5 *If $x \in ACP$ then there is no ν such that $\langle x, \nu \rangle \surd$.*

Proof Obvious. For immediate termination in HyPA, there must be a ϵ subterm of x. No other constants or operators introduce termination. ⊠

Lemma 6 *If $x \in ACP$ and there is a ν such that $\langle x, \nu \rangle \overset{a,\nu}{\to} \langle y, \nu \rangle$, (with $y \in$ HyPA) then either $y \leftrightarroweq \epsilon$ and $\langle x \rangle \xrightarrow{a} \surd$, or there exists $y' \in ACP$ such that $y \leftrightarroweq y'$ and $\langle x \rangle \xrightarrow{a} \langle y' \rangle$.*

Proof This proof uses induction on the structure of x. For $x \in$ ACP, we find the following cases.

1. $x = \delta$, which contradicts with the assumption that $\langle x, \nu \rangle \overset{a,\nu}{\to} \langle y, \nu \rangle$ for some $\nu \in Val$.
2. $x = a$, for which we find trivially $y = \epsilon$, and using the semantics of ACP $\langle x \rangle \xrightarrow{a} \surd$.
3. $x = x' \oplus x'' \ \wedge \ x', x'' \in$ ACP, for which we find one of the following hypotheses, using the semantics of HyPA.
   (a) $\langle x', \nu \rangle \overset{a,\nu}{\to} \langle y, \nu \rangle$ With induction, we find for x', one of the two following hypotheses
      i. $y \leftrightarroweq \epsilon \wedge \langle x' \rangle \xrightarrow{a} \surd$ From which we conclude, using the semantics of ACP, that $\langle x \rangle \xrightarrow{a} \surd$.
      ii. $\exists_{y'}\ y' \in \text{ACP} \wedge y \leftrightarroweq y' \ \wedge \langle x' \rangle \xrightarrow{a} \langle y' \rangle$ From which we conclude, using the semantics of ACP, that $\langle x \rangle \xrightarrow{a} \langle y' \rangle$.

   (b) $\langle x'', \nu \rangle \overset{a,\nu}{\to} \langle y, \nu \rangle$, similar to the previous case.
4. $x = x' \odot x'' \ \wedge \ x', x'' \in$ ACP, for which we find on of the following hypotheses, using the semantics of HyPA.
   (a) $\langle x', \nu \rangle \surd \ \wedge \ \langle x'', \nu \rangle \overset{a,\nu}{\to} \langle y, \nu \rangle$, which according to lemma 5 contradicts with the assumption that $x' \in ACP$.
   (b) $y = z \odot x'' \ \wedge \ \langle x', \nu \rangle \overset{a,\nu}{\to} \langle z, \nu \rangle$ With induction, we find for x', one of the two following hypotheses
      i. $z \leftrightarroweq \epsilon \wedge \langle x' \rangle \xrightarrow{a} \surd$ From which we conclude, using the semantics of ACP, that $\langle x \rangle \xrightarrow{a} \surd x''$, and using congruence of sequential composition together with the sound axiom $\epsilon \odot x'' \approx x''$, that $y \leftrightarroweq x''$.
      ii. $\exists_{z'}\ z' \in \text{ACP} \wedge z \leftrightarroweq z' \ \wedge \langle x' \rangle \xrightarrow{a} \langle z' \rangle$ From which we conclude, using the semantics of ACP, that $\langle x \rangle \xrightarrow{a} \langle z' \odot x'' \rangle$, and using congruence of the sequential composition that $y \leftrightarroweq z' \odot x''$.
5. $x = x' \| x'' \ \wedge \ x', x'' \in$ ACP, for which we find one of the following hypotheses, using the semantics of HyPA.
   (a) $\exists_z\ y = z \| x'' \ \wedge \ \langle x', \nu \rangle \overset{a,\nu}{\to} \langle z, \nu \rangle$. With induction, we find for x', one of the two following hypotheses

i. $z \leftrightarrow \epsilon \ \wedge \ \langle\, x' \,\rangle \overset{a}{\rightarrow} \surd$ From which we conclude that $\langle\, x \,\rangle \overset{a}{\rightarrow} \langle\, x'' \,\rangle$ and using equational reasoning $y \leftrightarrow x''$.

ii. $\exists_{z'} \ z' \in \text{ACP} \wedge z \leftrightarrow z' \ \wedge \ \langle\, x' \,\rangle \overset{a}{\rightarrow} \langle\, z' \,\rangle$ From which we conclude that $\langle\, x \,\rangle \overset{a}{\rightarrow} \langle\, z' \,\|\, x'' \,\rangle$ and using congruence $y \leftrightarrow z' \,\|\, x''$.

(b) $\exists_z \ y = x' \,\|\, z \ \wedge \ \langle\, x'', \nu \,\rangle \overset{a,\nu}{\rightarrow} \langle\, z, \nu \,\rangle$, which is similar to the previous case.

(c) $\exists_{a',a'',z,z'} \ y = z \,\|\, z' \wedge a = a'\gamma a'' \wedge \langle\, x', \nu \,\rangle \overset{a',\nu}{\rightarrow} \langle\, z, \nu \,\rangle \wedge \langle\, x'', \nu \,\rangle \overset{a'',\nu}{\rightarrow} \langle\, z', \nu \,\rangle$ With induction, we find for x', one of the four following hypotheses

i. $z \leftrightarrow \epsilon \ \wedge \ \langle\, x' \,\rangle \overset{a'}{\rightarrow} \surd \ \wedge \ z' \leftrightarrow \epsilon \ \wedge \ \langle\, x'' \,\rangle \overset{a''}{\rightarrow} \surd$ From which we conclude using the semantics of ACP that $\langle\, x \,\rangle \overset{a}{\rightarrow} \surd$, and using equational reasoning that $y \leftrightarrow \epsilon$.

ii. $z \leftrightarrow \epsilon \wedge \langle\, x' \,\rangle \overset{a'}{\rightarrow} \surd \wedge \exists_{w'} \ w' \in \text{ACP} \wedge z' \leftrightarrow w' \ \wedge \langle\, x'' \,\rangle \overset{a''}{\rightarrow} \langle\, w' \,\rangle$ From which we conclude using the semantics of ACP that $\langle\, x \,\rangle \overset{a}{\rightarrow} \langle\, w' \,\rangle$ and using congruence and (sound) equational reasoning that $y \leftrightarrow w'$.

iii. $\exists_w \ w \in \text{ACP} \wedge z \leftrightarrow w \ \wedge \ \langle\, x' \,\rangle \overset{a'}{\rightarrow} \langle\, w \,\rangle \ \wedge \ z' \leftrightarrow \epsilon \ \wedge \ \langle\, x'' \,\rangle \overset{a''}{\rightarrow} \surd$ Similar to the previous case.

iv. $\exists_{w,w'} \ w, w' \in \text{ACP} \wedge z \leftrightarrow w \wedge \langle\, x' \,\rangle \overset{a'}{\rightarrow} \langle\, w \,\rangle \wedge z' \leftrightarrow w' \wedge \langle\, x'' \,\rangle \overset{a''}{\rightarrow} \langle\, w' \,\rangle$ From which we conclude using the semantics of ACP that $\langle\, x \,\rangle \overset{a}{\rightarrow} \langle\, w \,\|\, w' \,\rangle$ and using congruence $y \leftrightarrow w \,\|\, w'$.

6. $x = x' \,\lfloor\!\lfloor\, x''$, which is a subcase of $x = x' \,\|\, x''$.

7. $x = x' \,|\, x''$, which is a subcase of $x = x' \,\|\, x''$.

8. $x = \partial_H\,(x') \wedge x' \in \text{ACP}$, for which we find one of the following hypotheses, using the semantics of HyPA.

(a) $\exists_z \ y = \partial_H\,(z) \ \wedge \ a \notin H \ \wedge \ \langle\, x', \nu \,\rangle \overset{a,\nu}{\rightarrow} \langle\, z, \nu \,\rangle$ With induction, we find for x', one of the two following hypotheses

i. $z \leftrightarrow \epsilon \wedge \langle\, x' \,\rangle \overset{a}{\rightarrow} \surd$ From which we conclude, using the semantics of ACP, that $\langle\, x \,\rangle \overset{a}{\rightarrow} \surd$.

ii. $\exists_{z'} \ z' \in \text{ACP} \wedge z \leftrightarrow z' \ \wedge \langle\, x' \,\rangle \overset{a}{\rightarrow} \langle\, z' \,\rangle$ From which we conclude, using the semantics of ACP, that $\langle\, x \,\rangle \overset{a}{\rightarrow} \langle\, \partial_{z'}\,() \rangle$ and using congruence $y \leftrightarrow \partial_H\,(z')$.

⊠

Using these four lemmas, we can prove the main theorem by showing that $\leftrightarrow$ is an ACP-bisimulation relation.

Corollary 1 *$\leftrightarrows$, restricted to closed ACP terms, is an ACP-bisimulation relation.*

Proof Suppose $p \leftrightarrows q$, and $p, q \in$ ACP.

- If $\langle p \rangle \xrightarrow{a}_{ACP} \langle p' \rangle$, then we use the lemma 4 to find $y \leftrightarrows p'$ such that $\langle p, \nu \rangle \overset{a,\nu}{\to} \langle y, \nu \rangle$ for every ν. Since $\leftrightarrows$ is a bisimulation relation, there exists $y' \leftrightarrows y$ such that $\langle q, \nu \rangle \overset{a,\nu}{\to} \langle y', \nu \rangle$. Using lemma 5, and the observation that $p' \in$ ACP, we find that not $\langle p', \nu \rangle \surd$. Now, we can use lemma 6 to find there exists a $q' \leftrightarrows y'$ such that $\langle q \rangle \xrightarrow{a}_{ACP} \langle q' \rangle$. Lastly, $\leftrightarrows$ is an equivalence relation, from which we conclude $p' \leftrightarrows q'$.

- If $\langle q \rangle \xrightarrow{a}_{ACP} \langle q' \rangle$, the reasoning is similar to the previous case.

- If $\langle p \rangle \xrightarrow{a}_{ACP} \surd$, then we use lemma 3 to find $y \leftrightarrows \epsilon$ such that $\langle p, \nu \rangle \overset{a,\nu}{\to} \langle y, \nu \rangle$ for every ν. Since $\leftrightarrows$ is a bisimulation relation, there exists $y' \leftrightarrows y$ such that $\langle q, \nu \rangle \overset{a,\nu}{\to} \langle y', \nu \rangle$. Now, using lemma 6 we may conclude that either, there exists $z \in$ ACP such that $z \leftrightarrows y'$ (which cannot be, since then $z \leftrightarrows \epsilon$ and $\langle z, \nu \rangle \surd$, contradicting lemma 4), or $y' \leftrightarrows \epsilon$ (which is true) and $\langle q \rangle \xrightarrow{a}_{ACP} \surd$.

- If $\langle q \rangle \xrightarrow{a}_{ACP} \surd$, the reasoning is similar to the previous case.

⊠

C Rewriting into basic terms

In this section, we will show that all the terms of HyPA can be rewritten into basic terms, using the axiomatization of HyPA. In fact, we show that there is a strongly normalizing rewrite system, for rewriting HyPA terms into the following form.

$$\begin{array}{rl} N' ::= & \delta \mid \epsilon \mid a \mid c \mid a \odot N' \mid c \rhd N' \mid d \gg a \mid d \gg c \mid \\ & d \gg \epsilon \mid d \gg a \odot N' \mid d \gg c \rhd N' \mid N' \oplus N' \ . \end{array}$$

After that, it is easy to verify that the terms in N' can be rewritten into N using the axioms $[\,\mathit{true}\,] \gg x \approx x \odot \epsilon \approx x \rhd \delta \approx x$ and $\delta \approx [\,\mathit{false}\,] \gg \epsilon$.

The remainder of this section consists of three parts. In part one, we give a rewrite system that is constructed for the task of rewriting HyPA terms into N'. Furthermore, we show that all the rewrite rules can be derived using the axiomatization of HyPA. And in part two, we show that all possible normal forms of the rewrite system are in N'. While in part three we show that the rewrite system is strongly normalizing, i.e. that every term has a normal form into which it can be rewritten.

C.1 The rewrite system

In this section, we give the rewrite system for rewriting HyPA terms into terms of the form N'. All the rules are derivable using the axiomatization, and are hence sound. This can be easily seen, since we have ordered the rules in groups, based on the most important axiom from which the rule is derived. Almost all rules are derivable using only the base axiom, and one or more of the following unit-, zero-, and commutativity-theorems. (Note that most of them are axioms.)

$x \odot \epsilon \approx x$	$\delta \odot x \approx \delta$	$\delta \lfloor\lfloor x \approx \delta$
$\delta \mid x \approx \delta$	$\delta \oplus x \approx x$	$[\,\mathit{true}\,] \gg x \approx x$
$[\,\mathit{false}\,] \gg x \approx \delta$	$x \rhd \delta \approx x$	$\epsilon \parallel x \approx x$
$x \mid y \approx y \mid x$	$x \oplus y \approx y \oplus x$	$x \parallel y \approx y \parallel x$

In the derivations, we have also used the following logical equivalences:

$d^{??} = d^{?}$	$d^{?} \wedge [\,\mathit{true}\,] = d^{?}$	$d \wedge d' = d' \wedge d$
$c \wedge c' = c' \wedge c$	$[\,\mathit{false}\,]^{?} = [\,\mathit{false}\,]$	$[\,\mathit{true}\,]^{?} = [\,\mathit{true}\,]$
$[\,\mathit{true}\,] \sim d = d$		

Only two rules are not derivable using only those axioms, and using the derivability of other rules. For those two rules, a proof is given in the end of the subsubsection in which they are introduced.

C.1.1 Rules from the axiom: $d \gg \delta \approx \delta$

$d \gg \delta \hookrightarrow \delta$

C.1.2 Rules from the axiom: $d \gg (d' \gg x) \approx (d \sim d') \gg x$

$d \gg d' \gg x \hookrightarrow (d \sim d') \gg x$

C.1.3 Rules from the axiom: $d \gg (x \oplus y) \approx d \gg x \oplus d \gg y$

$d \gg (x \oplus y) \hookrightarrow d \gg x \oplus d \gg y$

C.1.4 Rules from the axiom: $(x \odot y) \odot z \approx x \odot (y \odot z)$

$(x \odot y) \odot z \hookrightarrow x \odot (y \odot z)$

C.1.5 Rules from the axiom: $(x \oplus y) \odot z \approx x \odot z \oplus y \odot z$

$(x \oplus y) \odot z \hookrightarrow x \odot z \oplus y \odot z$

C.1.6 Rules from the axiom: $x \blacktriangleright y \approx x \triangleright y \oplus y$

$x \blacktriangleright y \hookrightarrow x \triangleright y \oplus y$

C.1.7 Rules from the axiom: $\epsilon \triangleright x \approx \epsilon$

$\epsilon \triangleright x \hookrightarrow \epsilon$

C.1.8 Rules from the axiom: $(x \triangleright y) \triangleright z \approx x \triangleright (y \blacktriangleright z)$

$(c \triangleright x) \triangleright y \hookrightarrow c \triangleright (x \blacktriangleright y)$

C.1.9 Rules from the axiom: $(x \oplus y) \triangleright z \approx x \triangleright z \oplus y \triangleright z$

$(x \oplus y) \triangleright z \hookrightarrow x \triangleright z \oplus y \triangleright z$

C.1.10 Rules from the axiom: $x \,\|\, y \approx x \,\lfloor\!\lfloor\, y \oplus y \,\lfloor\!\lfloor\, x \oplus x \mid y$

$x \,\|\, y \hookrightarrow x \,\lfloor\!\lfloor\, y \oplus y \,\lfloor\!\lfloor\, x \oplus x \mid y$

C.1.11 Rules from the axiom: $(x \oplus y) \,\lfloor\!\lfloor\, z \approx x \,\lfloor\!\lfloor\, z \oplus y \,\lfloor\!\lfloor\, z$

$(x \oplus y) \,\lfloor\!\lfloor\, z \hookrightarrow x \,\lfloor\!\lfloor\, z \oplus y \,\lfloor\!\lfloor\, z$

C.1.12 Rules from the axiom: $\epsilon \,\lfloor\!\lfloor\, x \approx \delta$

$\epsilon \,\lfloor\!\lfloor\, x \hookrightarrow \delta$

C.1.13 Rules from the axiom: $(d \gg \epsilon) \odot x \approx d^? \gg x$

$(d \gg \epsilon) \odot x \hookrightarrow d^? \gg x$
$\epsilon \odot x \hookrightarrow x$

C.1.14 Rules from the axiom: $(d \gg x) \mathbin{⌊\!⌊} y \approx d \gg (x \mathbin{⌊\!⌊} y)$

$(d \gg x) \mathbin{⌊\!⌊} y \hookrightarrow d \gg (x \mathbin{⌊\!⌊} y)$
$\delta \mathbin{⌊\!⌊} x \hookrightarrow \delta$

C.1.15 Rules from the axiom: $(a \odot x) \mathbin{⌊\!⌊} y \approx a \odot (x \parallel y)$

$(a \odot x) \mathbin{⌊\!⌊} y \hookrightarrow a \odot (x \parallel y)$
$a \mathbin{⌊\!⌊} x \hookrightarrow a \odot x$

C.1.16 Rules from the axiom: $(c \rhd x) \mathbin{⌊\!⌊} y \approx \delta$

$(c \rhd x) \mathbin{⌊\!⌊} y \hookrightarrow \delta$
$c \mathbin{⌊\!⌊} x \hookrightarrow \delta$

C.1.17 Rules from the axiom: $(d \gg a) \odot x \approx d \gg a \odot x$

$(d \gg a) \odot x \hookrightarrow d \gg a \odot x$
$\delta \odot x \hookrightarrow \delta$
$(d \gg a \odot x) \odot y \hookrightarrow d \gg a \odot (x \odot y)$

This last rule is perhaps not so trivial, and can be derived as follows:

$$\begin{aligned} (d \gg a \odot x) \odot y &\approx ((d \gg a) \odot x) \odot y \\ &\approx (d \gg a) \odot (x \odot y) \\ &\approx d \gg a \odot (x \odot y) \end{aligned}$$

C.1.18 Rules from the axiom: $(a \odot x) \rhd y \approx a \odot (x \blacktriangleright y)$

$(a \odot x) \rhd y \hookrightarrow a \odot (x \blacktriangleright y)$
$a \rhd x \hookrightarrow a \odot (\epsilon \oplus x)$

The derivation of this last rule goes as follows:

$$\begin{aligned} a \rhd x &\approx (a \odot \epsilon) \rhd x \\ &\approx a \odot (\epsilon \blacktriangleright x) \\ &\approx a \odot (\epsilon \rhd x \oplus x) \\ &\approx a \odot (\epsilon \oplus x) \end{aligned}$$

C.1.19 Rules from the axiom: $(d \gg x) \rhd y \approx d \gg x \rhd y$

$(d \gg x) \rhd y \hookrightarrow d \gg x \rhd y$
$\delta \rhd x \hookrightarrow \delta$

C.1.20 Rules from the axiom: $(d \gg c \rhd x) \odot y \approx d \gg c \rhd x \odot y$

$(d \gg c \rhd x) \odot y \hookrightarrow d \gg c \rhd x \odot y$
$(c \rhd x) \odot y \hookrightarrow c \rhd x \odot y$
$(d \gg c) \odot x \hookrightarrow d \gg c$
$c \odot x \hookrightarrow c$

C.1.21 Rules from the axiom: $\delta \,|\, x \approx \delta$

$\delta \,|\, x \hookrightarrow \delta$
$x \,|\, \delta \hookrightarrow \delta$

C.1.22 Rules from the axiom: $(x \oplus y) \,|\, z \approx x \,|\, z \oplus y \,|\, z$

$(x \oplus y) \,|\, z \hookrightarrow x \,|\, z \oplus y \,|\, z$
$x \,|\, (y \oplus z) \hookrightarrow x \,|\, y \oplus x \,|\, z$

C.1.23 Rules from the axiom: $d \gg \epsilon \,|\, d' \gg \epsilon \approx (d^? \wedge d'^?) \gg \epsilon$

$d \gg \epsilon \,|\, d' \gg \epsilon \hookrightarrow (d^? \wedge d'^?) \gg \epsilon$
$d \gg \epsilon \,|\, \epsilon \hookrightarrow d^? \gg \epsilon$
$\epsilon \,|\, d \gg \epsilon \hookrightarrow d^? \gg \epsilon$
$\epsilon \,|\, \epsilon \hookrightarrow \epsilon$

C.1.24 Rules from the axiom: $d \gg a \odot x \,|\, d' \gg a' \odot y \approx (d \wedge d') \gg (a \gamma a') \odot (x \,\|\, y)$ if $(a\gamma a')$ defined

For some of the rules below, it is important to notice that the reinitialization-clause $([\, \mathit{true} \,] \wedge d)$ is not logically equivalent to d, since $[\, \mathit{true} \,]$ prevents variables from changing their valuation. The unit element of $\wedge$ for reinitialization-clauses is $[\, \mathbb{V}_m \mid \mathit{true} \,]$.

$d \gg a \odot x \,|\, d' \gg a' \odot y \hookrightarrow (d \wedge d') \gg (a\gamma a') \odot (x \,\|\, y)$
$a \odot x \,|\, d \gg a' \odot y \hookrightarrow ([\, \mathit{true} \,] \wedge d) \gg (a\gamma a') \odot (x \,\|\, y)$
$a \odot x \,|\, a' \odot y \hookrightarrow (a\gamma a') \odot (x \,\|\, y)$
$d \gg a \odot x \,|\, a' \odot y \hookrightarrow (d \wedge [\, \mathit{true} \,]) \gg (a\gamma a') \odot (x \,\|\, y)$
$d \gg a \,|\, d' \gg a' \odot x \hookrightarrow (d \wedge d') \gg (a\gamma a') \odot x$
$d \gg a \odot x \,|\, d' \gg a' \hookrightarrow (d \wedge d') \gg (a\gamma a') \odot x$
$d \gg a \odot x \,|\, a' \hookrightarrow (d \wedge [\, \mathit{true} \,]) \gg (a\gamma a') \odot x$
$a \,|\, d \gg a' \odot x \hookrightarrow ([\, \mathit{true} \,] \wedge d) \gg (a\gamma a') \odot x$
$a \odot x \,|\, d \gg a' \hookrightarrow ([\, \mathit{true} \,] \wedge d) \gg (a\gamma a') \odot x$
$d \gg a \,|\, a' \odot x \hookrightarrow (d \wedge [\, \mathit{true} \,]) \gg (a\gamma a') \odot x$
$d \gg a \,|\, d' \gg a' \hookrightarrow (d \wedge d') \gg (a\gamma a')$
$d \gg a \,|\, a' \hookrightarrow (d \wedge [\, \mathit{true} \,]) \gg (a\gamma a')$
$a \,|\, d \gg a' \hookrightarrow ([\, \mathit{true} \,] \wedge d) \gg (a\gamma a')$
$a \,|\, a' \odot x \hookrightarrow (a\gamma a') \odot x$
$a \odot x \,|\, \epsilon \hookrightarrow (a\gamma a') \odot x$

$a \mid a' \hookrightarrow a\gamma a'$

C.1.25 Rules from the axiom: $d \gg a \odot x \mid d' \gg a' \odot y \approx \delta$ if $(a\gamma a')$ undefined

$d \gg a \odot x \mid d' \gg a' \odot y \hookrightarrow \delta$
$a \odot x \mid d \gg a' \odot y \hookrightarrow \delta$
$d \gg a \odot x \mid a' \odot y \hookrightarrow \delta$
$d \gg a \mid d' \gg a' \odot x \hookrightarrow \delta$
$d \gg a \odot x \mid d' \gg a' \hookrightarrow \delta$
$a \odot x \mid a' \odot y \hookrightarrow \delta$
$a \mid d \gg a' \odot x \hookrightarrow \delta$
$d \gg a \odot x \mid a' \hookrightarrow \delta$
$a \odot x \mid d \gg a' \hookrightarrow \delta$
$d \gg a \mid a' \odot x \hookrightarrow \delta$
$a \odot x \mid a' \hookrightarrow \delta$
$a \mid a' \odot x \hookrightarrow \delta$
$d \gg a \mid d' \gg a' \hookrightarrow \delta$
$a \mid d \gg a' \hookrightarrow \delta$
$d \gg a \mid a' \hookrightarrow \delta$
$a \mid a' \hookrightarrow \delta$

C.1.26 Rules from the axiom: $d \gg \epsilon \mid d' \gg (a \odot x) \approx \delta$

$d \gg \epsilon \mid d' \gg a \odot x \hookrightarrow \delta$
$d \gg a \odot x \mid d' \gg \epsilon \hookrightarrow \delta$
$\epsilon \mid d \gg a \odot x \hookrightarrow \delta$
$d \gg a \odot x \mid \epsilon \hookrightarrow \delta$
$a \odot x \mid d \gg \epsilon \hookrightarrow \delta$
$d \gg \epsilon \mid a \odot x \hookrightarrow \delta$
$d \gg a \mid d' \gg \epsilon \hookrightarrow \delta$
$d \gg \epsilon \mid d' \gg a \hookrightarrow \delta$
$\epsilon \mid a \odot x \hookrightarrow \delta$
$a \odot x \mid \epsilon \hookrightarrow \delta$
$\epsilon \mid d \gg a \hookrightarrow \delta$
$d \gg a \mid \epsilon \hookrightarrow \delta$
$d \gg \epsilon \mid a \hookrightarrow \delta$
$a \mid d \gg \epsilon \hookrightarrow \delta$
$\epsilon \mid a \hookrightarrow \delta$
$a \mid \epsilon \hookrightarrow \delta$

C.1.27 Rules from the axiom: $d \gg \epsilon \mid d' \gg c \rhd x \approx (d^? \sim d') \gg (c \rhd x)$

$d \gg \epsilon \mid d' \gg c \rhd x \hookrightarrow (d^? \sim d') \gg c \rhd x$
$d \gg c \rhd x \mid d' \gg \epsilon \hookrightarrow (d'^? \sim d) \gg c \rhd x$

$c \rhd x | d \gg \epsilon \hookrightarrow d^? \gg c \rhd x$
$d \gg \epsilon | c \rhd x \hookrightarrow d^? \gg c \rhd x$
$\epsilon | d \gg c \rhd x \hookrightarrow d \gg c \rhd x$
$d \gg c \rhd x | \epsilon \hookrightarrow d \gg c \rhd x$
$d \gg c | d' \gg \epsilon \hookrightarrow (d'^? \sim d) \gg c$
$d \gg \epsilon | d' \gg c \hookrightarrow (d^? \sim d') \gg c$
$\epsilon | c \rhd x \hookrightarrow c \rhd x$
$c \rhd x | \epsilon \hookrightarrow c \rhd x$
$c | d \gg \epsilon \hookrightarrow d^? \gg c$
$d \gg \epsilon | c \hookrightarrow d^? \gg c$
$\epsilon | d \gg c \hookrightarrow d \gg c$
$d \gg c | \epsilon \hookrightarrow d \gg c$
$\epsilon | c \hookrightarrow c$
$c | \epsilon \hookrightarrow c$

C.1.28 Rules from the axiom: $d \gg c \rhd x | d' \gg a \odot y \approx \delta$

$d \gg a \odot x | d' \gg c \rhd y \hookrightarrow \delta$
$d \gg c \rhd x | d' \gg a \odot y \hookrightarrow \delta$
$d \gg c \rhd x | a \odot x \hookrightarrow \delta$
$d \gg a \odot x | c \rhd y \hookrightarrow \delta$
$c \rhd x | d \gg a \odot y \hookrightarrow \delta$
$a \odot x | d \gg c \rhd y \hookrightarrow \delta$
$d \gg c | d' \gg a \odot x \hookrightarrow \delta$
$d \gg a \odot x | d' \gg c \hookrightarrow \delta$
$d \gg c \rhd x | d' \gg a \hookrightarrow \delta$
$d \gg a | d' \gg c \rhd x \hookrightarrow \delta$
$c \rhd x | a \odot y \hookrightarrow \delta$
$a \odot x | c \rhd y \hookrightarrow \delta$
$c | d \gg a \odot x \hookrightarrow \delta$
$a \odot x | d \gg c \hookrightarrow \delta$
$d \gg c | a \odot x \hookrightarrow \delta$
$d \gg a \odot x | c \hookrightarrow \delta$
$c \rhd x | d \gg a \hookrightarrow \delta$
$d \gg a | c \rhd x \hookrightarrow \delta$
$d \gg c \rhd x | a \hookrightarrow \delta$
$a | d \gg c \rhd x \hookrightarrow \delta$
$d \gg c | d' \gg a \hookrightarrow \delta$
$d \gg a | d \gg c \hookrightarrow \delta$
$a \odot x | c \hookrightarrow \delta$
$c | a \odot x \hookrightarrow \delta$
$a | c \rhd x \hookrightarrow \delta$
$c \rhd x | a \hookrightarrow \delta$
$a | d \gg c \hookrightarrow \delta$
$c | d \gg a \hookrightarrow \delta$

$d \gg a \mid c \hookrightarrow \delta$
$d \gg c \mid a \hookrightarrow \delta$
$a \mid c \hookrightarrow \delta$
$c \mid a \hookrightarrow \delta$

C.1.29 Rules from the axiom: $d \gg c \triangleright x \mid d' \gg c' \triangleright y \approx ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \triangleright (x \lfloor\!\lfloor c' \blacktriangleright y \oplus y \lfloor\!\lfloor c \blacktriangleright x \oplus x \mid c' \blacktriangleright y \oplus y \mid c \blacktriangleright x)$

$d \gg c \triangleright x \mid d' \gg c' \triangleright y \hookrightarrow$

$$((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \triangleright \begin{pmatrix} x \lfloor\!\lfloor c' \blacktriangleright y \oplus \\ y \lfloor\!\lfloor c \blacktriangleright x \oplus \\ x \mid c' \blacktriangleright y \oplus \\ y \mid c \blacktriangleright x \end{pmatrix}$$

$$c \triangleright x \mid d \gg c' \triangleright y \hookrightarrow (c_{jmp} \wedge (d \sim c'_{jmp})) \gg (c \wedge c') \triangleright \begin{pmatrix} x \lfloor\!\lfloor c' \blacktriangleright y \oplus \\ y \lfloor\!\lfloor c \blacktriangleright x \oplus \\ x \mid c' \blacktriangleright y \oplus \\ y \mid c \blacktriangleright x \end{pmatrix}$$

$$d \gg c \triangleright x \mid c' \triangleright y \hookrightarrow ((d \sim c_{jmp}) \wedge c'_{jmp}) \gg (c \wedge c') \triangleright \begin{pmatrix} x \lfloor\!\lfloor c' \blacktriangleright y \oplus \\ y \lfloor\!\lfloor c \blacktriangleright x \oplus \\ x \mid c' \blacktriangleright y \oplus \\ y \mid c \blacktriangleright x \end{pmatrix}$$

$$c \triangleright x \mid c' \triangleright y \hookrightarrow (c \wedge c')_{jmp} \gg (c \wedge c') \triangleright \begin{pmatrix} x \lfloor\!\lfloor c' \blacktriangleright y \oplus \\ y \lfloor\!\lfloor c \blacktriangleright x \oplus \\ x \mid c' \blacktriangleright y \oplus \\ y \mid c \blacktriangleright x \end{pmatrix}$$

$d \gg c \mid d' \gg c' \triangleright x \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \triangleright (c \parallel x)$
$d \gg c \triangleright x \mid d' \gg c' \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \triangleright (x \parallel c')$
$d \gg c \triangleright x \mid c' \hookrightarrow ((d \sim c_{jmp}) \wedge c'_{jmp}) \gg (c \wedge c') \triangleright (x \parallel c')$
$c \triangleright x \mid d \gg c' \hookrightarrow (c_{jmp} \wedge (d \sim c'_{jmp})) \gg (c \wedge c') \triangleright (x \parallel c')$
$d \gg c \mid c' \triangleright x \hookrightarrow ((d \sim c_{jmp}) \wedge c'_{jmp}) \gg (c \wedge c') \triangleright (c \parallel x)$
$c \mid d \gg c' \triangleright x \hookrightarrow (c_{jmp} \wedge (d \sim c'_{jmp})) \gg (c \wedge c') \triangleright (c \parallel x)$
$d \gg c \mid d' \gg c' \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c')$
$c \mid d \gg c' \hookrightarrow (c_{jmp} \wedge (d \sim c'_{jmp})) \gg (c \wedge c')$
$d \gg c \mid c' \hookrightarrow ((d \sim c_{jmp}) \wedge c'_{jmp}) \gg (c \wedge c')$
$c \triangleright x \mid c' \hookrightarrow (c \wedge c')_{jmp} \gg (c \wedge c') \triangleright (x \parallel c')$
$c \mid c' \triangleright x \hookrightarrow (c \wedge c')_{jmp} \gg (c \wedge c') \triangleright (c \parallel x)$
$c \mid c' \hookrightarrow (c \wedge c')_{jmp} \gg (c \wedge c')$

The rule $d \gg c \mid d' \gg c' \triangleright x \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \triangleright (c \parallel x)$ is not trivial. Therefore, we give its derivation.

$$
\begin{aligned}
d \gg c \,|\, d' \gg c' \triangleright x \quad &\approx \quad d \gg c \triangleright \delta \,|\, d' \gg c' \triangleright x \\
&\approx \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \triangleright \begin{pmatrix} \delta \lfloor\!\lfloor c' \blacktriangleright x \oplus \\ x \lfloor\!\lfloor c \blacktriangleright \delta \oplus \\ \delta \,|\, c' \blacktriangleright x \oplus \\ x \,|\, c \blacktriangleright \delta \end{pmatrix} \\
&\approx \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \triangleright \begin{pmatrix} \delta \oplus \\ x \lfloor\!\lfloor c \blacktriangleright \delta \oplus \\ \delta \oplus \\ x \,|\, c \blacktriangleright \delta \end{pmatrix} \\
&\approx \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \triangleright \begin{pmatrix} \delta \oplus \\ x \lfloor\!\lfloor (c \triangleright \delta \oplus \delta) \oplus \\ \delta \oplus \\ x \,|\, (c \triangleright \delta \oplus \delta) \end{pmatrix} \\
&\approx \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \triangleright \begin{pmatrix} x \lfloor\!\lfloor c \triangleright \delta \oplus \\ \delta \oplus \\ x \,|\, c \triangleright \delta \end{pmatrix} \\
&\approx \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \triangleright \begin{pmatrix} x \lfloor\!\lfloor c \triangleright \delta \oplus \\ c \triangleright \delta \lfloor\!\lfloor x \oplus \\ x \,|\, c \triangleright \delta \end{pmatrix} \\
&\approx \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \triangleright \begin{pmatrix} x \lfloor\!\lfloor c \oplus \\ c \lfloor\!\lfloor x \oplus \\ x \,|\, c \end{pmatrix} \\
&\approx \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \triangleright \begin{pmatrix} c \lfloor\!\lfloor x \oplus \\ x \lfloor\!\lfloor c \oplus \\ c \,|\, x \end{pmatrix} \\
&\approx \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \triangleright (c \,\|\, x)
\end{aligned}
$$

C.1.30 Rules from the axiom: $\partial_H(\epsilon) \approx \epsilon$

$\partial_H(\epsilon) \hookrightarrow \epsilon$

C.1.31 Rules from the axiom: if $a \in H$ then $\partial_H(a) \approx \delta$

if $a \in H$ then $\partial_H(a) \hookrightarrow \delta$

C.1.32 Rules from the axiom: if $a \notin H$ then $\partial_H(a) \approx a$

if $a \notin H$ then $\partial_H(a) \hookrightarrow a$

C.1.33 Rules from the axiom: $\partial_H(c) \approx c$

$\partial_H(c) \hookrightarrow c$

C.1.34 Rules from the axiom: $\partial_H(x \odot y) \approx \partial_H(x) \odot \partial_H(y)$

$\partial_H(x \odot y) \hookrightarrow \partial_H(x) \odot \partial_H(y)$

C.1.35 Rules from the axiom: $\partial_H(x \rhd y) \approx \partial_H(x) \rhd \partial_H(y)$

$\partial_H(x \rhd y) \hookrightarrow \partial_H(x) \rhd \partial_H(y)$

C.1.36 Rules from the axiom: $\partial_H(d \gg x) \approx d \gg \partial_H(x)$

$\partial_H(d \gg x) \hookrightarrow d \gg \partial_H(x)$
$\partial_H(\delta) \hookrightarrow \delta$

C.1.37 Rules from the axiom: $\partial_H((x \oplus y)) \approx \partial_H(x) \oplus \partial_H(y)$

$\partial_H(x \oplus y) \hookrightarrow \partial_H(x) \oplus \partial_H(y)$

C.2 Normal forms are in N'

Since the atoms of HyPA are also atoms of N', every HyPA term $p \notin N'$ has a subterm $s \notin N'$ of the form:

$$\begin{array}{rcl} P & ::= & d \gg N' \mid N' \oplus N' \mid N' \odot N' \mid N' \blacktriangleright N' \mid N' \rhd N' \mid \\ & & N' \parallel N' \mid N' \mathbin{\lfloor\!\lfloor} N' \mid N' \mid N' \mid \partial_H(N') \end{array}$$

In the following paragraphs, we will give one or more applicable rewrite rules for every of these possible subterms, unless the specific subterm is itself in normal form. In that case, we do not need to give a rule since we have a contradiction with the assumption that $s \notin N'$.

C.2.1 The form: $s \in d \gg N'$

We find the following cases:

- $s \in d \gg \delta$, which rewrites using $d \gg \delta \hookrightarrow \delta$;
- $s \in d \gg \epsilon \in N'$;
- $s \in d \gg a \in N'$;
- $s \in d \gg c \in N'$;

- $s \in d \gg a \odot N' \in N'$;
- $s \in d \gg c \rhd N' \in N'$;
- $s \in d \gg d' \gg a$, which rewrites using $d \gg d' \gg x \hookrightarrow (d \sim d') \gg x$;
- $s \in d \gg d' \gg c$, which rewrites using $d \gg d' \gg x \hookrightarrow (d \sim d') \gg x$;
- $s \in d \gg d' \gg \epsilon$, which rewrites using $d \gg d' \gg x \hookrightarrow (d \sim d') \gg x$;
- $s \in d \gg d' \gg a \odot N'$, which rewrites using $d \gg d' \gg x \hookrightarrow (d \sim d') \gg x$;
- $s \in d \gg d' \gg c \rhd N'$, which rewrites using $d \gg d' \gg x \hookrightarrow (d \sim d') \gg x$;
- $s \in d \gg (N' \oplus N')$, which rewrites using $d \gg (x \oplus y) \hookrightarrow d \gg x \oplus d \gg y$.

C.2.2 The form: $s \in N' \oplus N'$

Which directly leads to $s \in N'$.

C.2.3 The form: $s \in N' \odot N'$

We find the following cases

- $s \in \delta \odot N'$, which rewrites using $\delta \odot x \hookrightarrow \delta$;
- $s \in \epsilon \odot N'$, which rewrites using $\epsilon \odot x \hookrightarrow x$;
- $s \in a \odot N' \subseteq N'$;
- $s \in c \odot N'$, which rewrites using $c \odot x \hookrightarrow c$;
- $s \in (a \odot N') \odot N'$, which rewrites using $(x \odot y) \odot z \hookrightarrow x \odot (y \odot z)$;
- $s \in (c \rhd N') \odot N'$, which rewrites using $(c \rhd x) \odot y \hookrightarrow c \rhd x \odot y$;
- $s \in (d \gg a) \odot N'$, which rewrites using $(d \gg a) \odot x \hookrightarrow d \gg a \odot x$;
- $s \in (d \gg c) \odot N'$, which rewrites using $(d \gg c) \odot x \hookrightarrow d \gg c$;
- $s \in (d \gg \epsilon) \odot N'$, which rewrites using $(d \gg \epsilon) \odot x \hookrightarrow d^? \gg x$;
- $s \in (d \gg a \odot N') \odot N'$, which rewrites using $(d \gg a \odot x) \odot y \hookrightarrow d \gg a \odot (x \odot y)$;
- $s \in (d \gg c \rhd N') \odot N'$, which rewrites using $(d \gg c \rhd x) \odot y \hookrightarrow d \gg c \rhd x \odot y$;
- $s \in (N' \oplus N') \odot N'$, which rewrites using $(x \oplus y) \odot z \hookrightarrow x \odot z \oplus y \odot z$.

C.2.4 The form: $s \in N' \blacktriangleright N'$

Which rewrites using $x \blacktriangleright y \hookrightarrow x \rhd y \oplus y$.

C.2.5 The form: $s \in N' \rhd N'$

We find the following cases

- $s \in \delta \rhd N'$, which rewrites using $\delta \rhd x \hookrightarrow \delta$;
- $s \in \epsilon \rhd N'$, which rewrites using $\epsilon \rhd x \hookrightarrow \epsilon$;
- $s \in a \rhd N'$, which rewrites using $a \rhd x \hookrightarrow a \odot (\epsilon \oplus x)$;
- $s \in c \rhd N' \subseteq N'$;
- $s \in (a \odot N') \rhd N'$, which rewrites using $(a \odot x) \rhd y \hookrightarrow a \odot (x \blacktriangleright y)$;
- $s \in (c \rhd N') \rhd N'$, which rewrites using $(c \rhd x) \rhd y \hookrightarrow c \rhd (x \blacktriangleright y)$;
- $s \in (d \gg a) \rhd N'$, which rewrites using $(d \gg x) \rhd y \hookrightarrow d \gg x \rhd y$;
- $s \in (d \gg c) \rhd N'$, which rewrites using $(d \gg x) \rhd y \hookrightarrow d \gg x \rhd y$;
- $s \in (d \gg \epsilon) \rhd N'$, which rewrites using $(d \gg x) \rhd y \hookrightarrow d \gg x \rhd y$;
- $s \in (d \gg a \odot N') \rhd N'$, which rewrites using $(d \gg x) \rhd y \hookrightarrow d \gg x \rhd y$;
- $s \in (d \gg c \rhd N') \rhd N'$, which rewrites using $(d \gg x) \rhd y \hookrightarrow d \gg x \rhd y$;
- $s \in (N' \oplus N') \rhd N'$, which rewrites using $(x \oplus y) \rhd z \hookrightarrow x \rhd z \oplus y \rhd z$.

C.2.6 The form: $s \in N' \,\|\, N'$

Which rewrites using $x \,\|\, y \hookrightarrow x \,⌊\!⌊\, y \oplus y \,⌊\!⌊\, x \oplus x \mid y$.

C.2.7 The form: $s \in N' \,⌊\!⌊\, N'$

We find the following cases

- $s \in \delta \,⌊\!⌊\, N'$, which rewrites using $\delta \,⌊\!⌊\, x \hookrightarrow \delta$;
- $s \in \epsilon \,⌊\!⌊\, N'$, which rewrites using $\epsilon \,⌊\!⌊\, x \hookrightarrow \delta$;
- $s \in a \,⌊\!⌊\, N'$, which rewrites using $a \,⌊\!⌊\, x \hookrightarrow a \odot x$;
- $s \in c \,⌊\!⌊\, N'$, which rewrites using $c \,⌊\!⌊\, x \hookrightarrow \delta$;
- $s \in (a \odot N') \,⌊\!⌊\, N'$, which rewrites using $(a \odot x) \,⌊\!⌊\, y \hookrightarrow a \odot (x \,\|\, y)$;
- $s \in (c \rhd N') \,⌊\!⌊\, N'$, which rewrites using $(c \rhd x) \,⌊\!⌊\, y \hookrightarrow \delta$;
- $s \in (d \gg a) \,⌊\!⌊\, N'$, which rewrites using $(d \gg x) \,⌊\!⌊\, y \hookrightarrow d \gg x \,⌊\!⌊\, y$;
- $s \in (d \gg c) \,⌊\!⌊\, N'$, which rewrites using $(d \gg x) \,⌊\!⌊\, y \hookrightarrow d \gg x \,⌊\!⌊\, y$;

- $s \in (d \gg \epsilon) \lfloor\!\lfloor N'$, which rewrites using $(d \gg x) \lfloor\!\lfloor y \hookrightarrow d \gg x \lfloor\!\lfloor y$;
- $s \in (d \gg a \odot N') \lfloor\!\lfloor N'$, which rewrites using $(d \gg x) \lfloor\!\lfloor y \hookrightarrow d \gg x \lfloor\!\lfloor y$;
- $s \in (d \gg c \rhd N') \lfloor\!\lfloor N'$, which rewrites using $(d \gg x) \lfloor\!\lfloor y \hookrightarrow d \gg x \lfloor\!\lfloor y$;
- $s \in (N' \oplus N') \lfloor\!\lfloor N'$, which rewrites using $(x \oplus y) \lfloor\!\lfloor z \hookrightarrow x \lfloor\!\lfloor z \oplus y \lfloor\!\lfloor z$.

C.2.8 The form: $s \in N' \mid N'$

We find the following cases

- $s \in \delta \mid N'$, which rewrites using $\delta \mid x \hookrightarrow \delta$;
- $s \in \epsilon \mid N'$, for which we find the cases
  - $s \in \epsilon \mid \delta$, which rewrites using $x \mid \delta \hookrightarrow \delta$;
  - $s \in \epsilon \mid \epsilon$, which rewrites using $\epsilon \mid \epsilon \hookrightarrow \epsilon$;
  - $s \in \epsilon \mid a$, which rewrites using $\epsilon \mid a \hookrightarrow \delta$;
  - $s \in \epsilon \mid c$, which rewrites using $\epsilon \mid c \hookrightarrow c$;
  - $s \in \epsilon \mid a \odot N'$, which rewrites using $\epsilon \mid a \odot x \hookrightarrow \delta$;
  - $s \in \epsilon \mid c \rhd N'$, which rewrites using $\epsilon \mid c \rhd x \hookrightarrow c \rhd x$;
  - $s \in \epsilon \mid d \gg a$, which rewrites using $\epsilon \mid d \gg a \hookrightarrow \delta$;
  - $s \in \epsilon \mid d \gg c$, which rewrites using $\epsilon \mid d \gg c \hookrightarrow d \gg c$;
  - $s \in \epsilon \mid d \gg \epsilon$, which rewrites using $\epsilon \mid d \gg \epsilon \hookrightarrow d^? \gg \epsilon$;
  - $s \in \epsilon \mid d \gg a \odot N'$, which rewrites using $\epsilon \mid d \gg a \odot x \hookrightarrow \delta$;
  - $s \in \epsilon \mid d \gg c \rhd N'$, which rewrites using $\epsilon \mid d \gg c \rhd x \hookrightarrow d \gg c \rhd x$;
  - $s \in \epsilon \mid (N' \oplus N')$, which rewrites using $x \mid (y \oplus z) \hookrightarrow x \mid y \oplus x \mid z$.
- $s \in a \mid N'$, for which we find the cases
  - $s \in a \mid \delta$, which rewrites using $x \mid \delta \hookrightarrow \delta$;
  - $s \in a \mid \epsilon$, which rewrites using $a \mid \epsilon \hookrightarrow \delta$;
  - $s \in a \mid a'$, for which we find two cases
    - If $a\gamma a'$ defined then we rewrite using $a \mid a' \hookrightarrow a\gamma a'$;
    - If $a\gamma a'$ undefined then we rewrite using $a \mid a' \hookrightarrow \delta$;
  - $s \in a \mid c$, which we rewrite using $a \mid c \hookrightarrow \delta$;
  - $s \in a \mid a' \odot N'$, for which we find two cases
    - If $a\gamma a'$ defined then we rewrite using $a \mid a' \odot x \hookrightarrow (a\gamma a') \odot x$;
    - If $a\gamma a'$ undefined then we rewrite using $a \mid a' \odot x \hookrightarrow \delta$;
  - $s \in a \mid c \rhd N'$, which we rewrite using $a \mid c \rhd x \hookrightarrow \delta$;
  - $s \in a \mid d \gg a'$, for which we find two cases

    * If $a\gamma a'$ defined then we rewrite using $a \,|\, d \gg a' \hookrightarrow ([\,true\,] \wedge d) \gg (a\gamma a')$;
    * If $a\gamma a'$ undefined then we rewrite using $a \,|\, d \gg a' \hookrightarrow \delta$;
  - $s \in a \,|\, d \gg c$, which we rewrite using $a \,|\, d \gg c \hookrightarrow \delta$;
  - $s \in a \,|\, d \gg \epsilon$, which we rewrite using $a \,|\, d \gg \epsilon \hookrightarrow \delta$;
  - $s \in a \,|\, d \gg a' \odot N'$, for which we find two cases
    * If $a\gamma a'$ defined then we rewrite using $a \,|\, d \gg a' \odot x \hookrightarrow ([\,true\,] \wedge d) \gg (a\gamma a') \odot x$;
    * If $a\gamma a'$ undefined then we rewrite using $a \,|\, d \gg a' \odot x \hookrightarrow \delta$;
  - $s \in a \,|\, d \gg c \rhd N'$, which we rewrite using $a \,|\, d \gg c \rhd x \hookrightarrow \delta$;
  - $s \in a \,|\, (N' \oplus N')$, which rewrites using $x \,|\, (y \oplus z) \hookrightarrow x \,|\, y \oplus x \,|\, z$.

- $s \in c \,|\, N'$, for which we find the cases
  - $s \in c \,|\, \delta$, which rewrites using $x \,|\, \delta \hookrightarrow \delta$;
  - $s \in c \,|\, \epsilon$, which rewrites using $c \,|\, \epsilon \hookrightarrow c$;
  - $s \in c \,|\, a$, which rewrites using $c \,|\, a \hookrightarrow \delta$;
  - $s \in c \,|\, c'$, which rewrites using $c \,|\, c' \hookrightarrow (c \wedge c')_{jmp} \gg (c \wedge c')$;
  - $s \in c \,|\, a \odot N'$, which rewrites using $c \,|\, a \odot x \hookrightarrow \delta$;
  - $s \in c \,|\, c' \rhd N'$, which rewrites using $c \,|\, c' \rhd x \hookrightarrow (c \wedge c')_{jmp} \gg (c \wedge c') \rhd (c \,\|\, x)$;
  - $s \in c \,|\, d \gg a$, which rewrites using $c \,|\, d \gg a \hookrightarrow \delta$;
  - $s \in c \,|\, d \gg c'$, which rewrites using $c \,|\, d \gg c' \hookrightarrow (c_{jmp} \wedge (d \sim c'_{jmp})) \gg (c \wedge c')$;
  - $s \in c \,|\, d \gg \epsilon$, which rewrites using $c \,|\, d \gg \epsilon \hookrightarrow d^? \gg c$;
  - $s \in c \,|\, d \gg a \odot N'$, which rewrites using $c \,|\, d \gg a \odot x \hookrightarrow \delta$;
  - $s \in c \,|\, d \gg c' \rhd N'$, which rewrites using $c \,|\, d \gg c' \rhd x \hookrightarrow (c_{jmp} \wedge (d \sim c'_{jmp})) \gg (c \wedge c') \rhd (c \,\|\, x)$;
  - $s \in c \,|\, (N' \oplus N')$, which rewrites using $x \,|\, (y \oplus z) \hookrightarrow x \,|\, y \oplus x \,|\, z$.

- $s \in a \odot N' \,|\, N'$, for which we find the cases
  - $s \in a \odot N' \,|\, \delta$, which rewrites using $x \,|\, \delta \hookrightarrow \delta$;
  - $s \in a \odot N' \,|\, \epsilon$, which rewrites using $a \odot x \,|\, \epsilon \hookrightarrow \delta$;
  - $s \in a \odot N' \,|\, a'$, for which we find two cases
    * If $a\gamma a'$ defined then we rewrite using $a \odot x \,|\, a' \hookrightarrow (a\gamma a') \odot x$;
    * If $a\gamma a'$ undefined then we rewrite using $a \odot x \,|\, a' \hookrightarrow \delta$;
  - $s \in a \odot N' \,|\, c$, which rewrites using $a \odot x \,|\, c \hookrightarrow \delta$;
  - $s \in a \odot N' \,|\, a' \odot N'$, for which we find two cases

    * If $a\gamma a'$ defined then we rewrite using $a \odot x \,|\, a' \odot y \hookrightarrow (a\gamma a') \odot (x \,\|\, y)$;
    * If $a\gamma a'$ undefined then we rewrite using $a \odot x \,|\, a' \odot y \hookrightarrow \delta$;
  - $s \in a \odot N' \,|\, c \rhd N'$, which rewrites using $a \odot x \,|\, c \rhd y \hookrightarrow \delta$;
  - $s \in a \odot N' \,|\, d \gg a'$, for which we find two cases
    * If $a\gamma a'$ defined then we rewrite using $a \odot x \,|\, d \gg a' \hookrightarrow ([\,\mathit{true}\,] \wedge d) \gg (a\gamma a') \odot x$;
    * If $a\gamma a'$ undefined then we rewrite using $a \odot x \,|\, d \gg a' \hookrightarrow \delta$;
  - $s \in a \odot N' \,|\, d \gg c$, which rewrites using $a \odot x \,|\, d \gg c \hookrightarrow \delta$;
  - $s \in a \odot N' \,|\, d \gg \epsilon$, which rewrites using $a \odot x \,|\, d \gg \epsilon \hookrightarrow \delta$;
  - $s \in a \odot N' \,|\, d \gg a' \odot N'$, for which we find two cases
    * If $a\gamma a'$ defined then we rewrite using $a \odot x \,|\, d \gg a' \odot y \hookrightarrow ([\,\mathit{true}\,] \wedge d) \gg (a\gamma a') \odot (x \,\|\, y)$;
    * If $a\gamma a'$ undefined then we rewrite using $a \odot x \,|\, d \gg a' \odot y \hookrightarrow \delta$;
  - $s \in a \odot N' \,|\, d \gg c \rhd N'$, which rewrites using $a \odot x \,|\, d \gg c \rhd y \hookrightarrow \delta$;
  - $s \in a \odot N' \,|\, (N' \oplus N')$, which rewrites using $x \,|\, (y \oplus z) \hookrightarrow x \,|\, y \oplus x \,|\, z$.

- $s \in c \rhd N' \,|\, N'$, for which we find the cases
  - $s \in c \rhd N' \,|\, \delta$, which rewrites using $x \,|\, \delta \hookrightarrow \delta$;
  - $s \in c \rhd N' \,|\, \epsilon$, which rewrites using $c \rhd x \,|\, \epsilon \hookrightarrow c \rhd x$;
  - $s \in c \rhd N' \,|\, a$, which rewrites using $c \rhd x \,|\, a \hookrightarrow \delta$;
  - $s \in c \rhd N' \,|\, c'$, which rewrites using $c \rhd x \,|\, c' \hookrightarrow (c \wedge c')_{jmp} \gg (c \wedge c') \rhd (x \,\|\, c')$;
  - $s \in c \rhd N' \,|\, a \odot N'$, which rewrites using $c \rhd x \,|\, a \odot y \hookrightarrow \delta$;
  - $s \in c \rhd N' \,|\, c' \rhd N'$, which rewrites using $c \rhd x \,|\, c' \rhd y \hookrightarrow$ $(c \wedge c')_{jmp} \gg (c \wedge c') \rhd \begin{pmatrix} x \,\lfloor\!\lfloor\, c' \blacktriangleright y \,\oplus \\ y \,\lfloor\!\lfloor\, c \blacktriangleright x \,\oplus \\ x \,|\, c' \blacktriangleright y \,\oplus \\ y \,|\, c \blacktriangleright x \end{pmatrix}$;
  - $s \in c \rhd N' \,|\, d \gg a$, which rewrites using $c \rhd x \,|\, d \gg a \hookrightarrow \delta$;
  - $s \in c \rhd N' \,|\, d \gg c'$, which rewrites using $c \rhd x \,|\, d \gg c' \hookrightarrow (c_{jmp} \wedge (d \sim c'_{jmp})) \gg (c \wedge c') \rhd (x \,\|\, c')$;
  - $s \in c \rhd N' \,|\, d \gg \epsilon$, which rewrites using $c \rhd x \,|\, d \gg \epsilon \hookrightarrow d^? \gg c \rhd x$;
  - $s \in c \rhd N' \,|\, d \gg a \odot N'$, which rewrites using $c \rhd x \,|\, d \gg a \odot y \hookrightarrow \delta$;

- $s \in c \rhd N' \,|\, d \gg c' \rhd N'$, which rewrites using $c \rhd x \,|\, d \gg c' \rhd y \hookrightarrow (c_{jmp} \wedge (d \sim c'_{jmp})) \gg (c \wedge c') \rhd \begin{pmatrix} x \lfloor\!\lfloor c' \blacktriangleright y \oplus \\ y \lfloor\!\lfloor c \blacktriangleright x \oplus \\ x \,|\, c' \blacktriangleright y \oplus \\ y \,|\, c \blacktriangleright x \end{pmatrix}$;
- $s \in c \rhd N' \,|\, (N' \oplus N')$, which rewrites using $x \,|\, (y \oplus z) \hookrightarrow x \,|\, y \oplus x \,|\, z$.

- $s \in d \gg a \,|\, N'$, for which we find the cases
  - $s \in d \gg a \,|\, \delta$, which rewrites using $x \,|\, \delta \hookrightarrow \delta$;
  - $s \in d \gg a \,|\, \epsilon$, which rewrites using $d \gg a \,|\, \epsilon \hookrightarrow \delta$;
  - $s \in d \gg a \,|\, a'$, for which we find two cases
    - If $a \gamma a'$ defined then we rewrite using $d \gg a \,|\, a' \hookrightarrow (d \wedge [\,\mathit{true}\,]) \gg (a \gamma a')$;
    - If $a \gamma a'$ undefined then we rewrite using $d \gg a \,|\, a' \hookrightarrow \delta$;
  - $s \in d \gg a \,|\, c$, which rewrites using $d \gg a \,|\, c \hookrightarrow \delta$;
  - $s \in d \gg a \,|\, a' \odot N'$, for which we find two cases
    - If $a \gamma a'$ defined then we rewrite using $d \gg a \,|\, a' \odot x \hookrightarrow (d \wedge [\,\mathit{true}\,]) \gg (a \gamma a') \odot x$;
    - If $a \gamma a'$ undefined then we rewrite using $d \gg a \,|\, a' \odot x \hookrightarrow \delta$;
  - $s \in d \gg a \,|\, c \rhd N'$, which rewrites using $d \gg a \,|\, c \rhd x \hookrightarrow \delta$;
  - $s \in d \gg a \,|\, d' \gg a'$, for which we find two cases
    - If $a \gamma a'$ defined then we rewrite using $d \gg a \,|\, d' \gg a' \hookrightarrow (d \wedge d') \gg (a \gamma a')$;
    - If $a \gamma a'$ undefined then we rewrite using $d \gg a \,|\, d' \gg a' \hookrightarrow \delta$;
  - $s \in d \gg a \,|\, d' \gg c$, which rewrites using $d \gg a \,|\, d \gg c \hookrightarrow \delta$;
  - $s \in d \gg a \,|\, d' \gg \epsilon$, which rewrites using $d \gg a \,|\, d' \gg \epsilon \hookrightarrow \delta$;
  - $s \in d \gg a \,|\, d' \gg a' \odot N'$, for which we find two cases
    - If $a \gamma a'$ defined then we rewrite using $d \gg a \,|\, d' \gg a' \odot x \hookrightarrow (d \wedge d') \gg (a \gamma a') \odot x$;
    - If $a \gamma a'$ undefined then we rewrite using $d \gg a \,|\, d' \gg a' \odot x \hookrightarrow \delta$;
  - $s \in d \gg a \,|\, d' \gg c \rhd N'$, which rewrites using $d \gg a \,|\, d' \gg c \rhd x \hookrightarrow \delta$;
  - $s \in d \gg a \,|\, (N' \oplus N')$, which rewrites using $x \,|\, (y \oplus z) \hookrightarrow x \,|\, y \oplus x \,|\, z$.

- $s \in d \gg c \,|\, N'$, for which we find the cases
  - $s \in d \gg c \,|\, \delta$, which rewrites using $x \,|\, \delta \hookrightarrow \delta$;
  - $s \in d \gg c \,|\, \epsilon$, which rewrites using $d \gg c \,|\, \epsilon \hookrightarrow d \gg c$;
  - $s \in d \gg c \,|\, a$, which rewrites using $d \gg c \,|\, a \hookrightarrow \delta$;

- $s \in d \gg c \,|\, c'$, which rewrites using $d \gg c \,|\, c' \hookrightarrow ((d \sim c_{jmp}) \wedge c'_{jmp}) \gg (c \wedge c')$;
- $s \in d \gg c \,|\, a \odot N'$, which rewrites using $d \gg c \,|\, a \odot x \hookrightarrow \delta$;
- $s \in d \gg c \,|\, c' \rhd N'$, which rewrites using $d \gg c \,|\, c' \rhd x \hookrightarrow ((d \sim c_{jmp}) \wedge c'_{jmp}) \gg (c \wedge c') \rhd (c \,||\, x)$;
- $s \in d \gg c \,|\, d' \gg a$, which rewrites using $d \gg c \,|\, d' \gg a \hookrightarrow \delta$;
- $s \in d \gg c \,|\, d' \gg c'$, which rewrites using $d \gg c \,|\, d' \gg c' \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c')$;
- $s \in d \gg c \,|\, d' \gg \epsilon$, which rewrites using $d \gg c \,|\, d' \gg \epsilon \hookrightarrow (d'^{?} \sim d) \gg c$;
- $s \in d \gg c \,|\, d' \gg a \odot N'$, which rewrites using $d \gg c \,|\, d' \gg a \odot x \hookrightarrow \delta$;
- $s \in d \gg c \,|\, d' \gg c' \rhd N'$, which rewrites using $d \gg c \,|\, d' \gg c' \rhd x \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (c \,||\, x)$;
- $s \in d \gg c \,|\, (N' \oplus N')$, which rewrites using $x \,|\, (y \oplus z) \hookrightarrow x \,|\, y \oplus x \,|\, z$.

- $s \in d \gg \epsilon \,|\, N'$, for which we find the cases
  - $s \in d \gg \epsilon \,|\, \delta$, which rewrites using $x \,|\, \delta \hookrightarrow \delta$;
  - $s \in d \gg \epsilon \,|\, \epsilon$, which rewrites using $d \gg \epsilon \,|\, \epsilon \hookrightarrow d^{?} \gg \epsilon$;
  - $s \in d \gg \epsilon \,|\, a$, which rewrites using $d \gg \epsilon \,|\, a \hookrightarrow \delta$;
  - $s \in d \gg \epsilon \,|\, c$, which rewrites using $d \gg \epsilon \,|\, c \hookrightarrow d^{?} \gg c$;
  - $s \in d \gg \epsilon \,|\, a \odot N'$, which rewrites using $d \gg \epsilon \,|\, a \odot x \hookrightarrow \delta$;
  - $s \in d \gg \epsilon \,|\, c \rhd N'$, which rewrites using $d \gg \epsilon \,|\, c \rhd x \hookrightarrow d^{?} \gg c \rhd x$;
  - $s \in d \gg \epsilon \,|\, d' \gg a$, which rewrites using $d \gg \epsilon \,|\, d' \gg a \hookrightarrow \delta$;
  - $s \in d \gg \epsilon \,|\, d' \gg c$, which rewrites using $d \gg \epsilon \,|\, d' \gg c \hookrightarrow (d^{?} \sim d') \gg c$;
  - $s \in d \gg \epsilon \,|\, d' \gg \epsilon$, which rewrites using $d \gg \epsilon \,|\, d' \gg \epsilon \hookrightarrow (d^{?} \wedge d'^{?}) \gg \epsilon$;
  - $s \in d \gg \epsilon \,|\, d' \gg a \odot N'$, which rewrites using $d \gg \epsilon \,|\, d' \gg a \odot x \hookrightarrow \delta$;
  - $s \in d \gg \epsilon \,|\, d' \gg c \rhd N'$, which rewrites using $d \gg \epsilon \,|\, d' \gg c \rhd x \hookrightarrow (d^{?} \sim d') \gg c \rhd x$;
  - $s \in d \gg \epsilon \,|\, (N' \oplus N')$, which rewrites using $x \,|\, (y \oplus z) \hookrightarrow x \,|\, y \oplus x \,|\, z$.
- $s \in d \gg a \odot N' \,|\, N'$, for which we find the cases
  - $s \in d \gg a \odot N' \,|\, \delta$, which rewrites using $x \,|\, \delta \hookrightarrow \delta$;
  - $s \in d \gg a \odot N' \,|\, \epsilon$, which rewrites using $d \gg a \odot x \,|\, \epsilon \hookrightarrow \delta$;

  - $s \in d \gg a \odot N' \mid a'$, for which we find two cases
    * If $a\gamma a'$ defined then we rewrite using $d \gg a \odot x \mid a' \hookrightarrow (d \wedge [\mathit{true}]) \gg (a\gamma a') \odot x$;
    * If $a\gamma a'$ undefined then we rewrite using $d \gg a \odot x \mid a' \hookrightarrow \delta$;
  - $s \in d \gg a \odot N' \mid c$, which rewrites using $d \gg a \odot x \mid c \hookrightarrow \delta$;
  - $s \in d \gg a \odot N' \mid a' \odot N'$, for which we find two cases
    * If $a\gamma a'$ defined then we rewrite using $d \gg a \odot x \mid a' \odot y \hookrightarrow (d \wedge [\mathit{true}]) \gg (a\gamma a') \odot (x \parallel y)$;
    * If $a\gamma a'$ undefined then we rewrite using $d \gg a \odot x \mid a' \odot y \hookrightarrow \delta$;
  - $s \in d \gg a \odot N' \mid c \rhd N'$, which rewrites using $d \gg a \odot x \mid c \rhd y \hookrightarrow \delta$;
  - $s \in d \gg a \odot N' \mid d' \gg a'$, for which we find two cases
    * If $a\gamma a'$ defined then we rewrite using $d \gg a \odot x \mid d' \gg a' \hookrightarrow (d \wedge d') \gg (a\gamma a') \odot x$;
    * If $a\gamma a'$ undefined then we rewrite using $d \gg a \odot x \mid d' \gg a' \hookrightarrow \delta$;
  - $s \in d \gg a \odot N' \mid d' \gg c$, which rewrites using $d \gg a \odot x \mid d' \gg c \hookrightarrow \delta$;
  - $s \in d \gg a \odot N' \mid d' \gg \epsilon$, which rewrites using $d \gg a \odot x \mid d' \gg \epsilon \hookrightarrow \delta$;
  - $s \in d \gg a \odot N' \mid d' \gg a' \odot N'$, for which we find two cases
    * If $a\gamma a'$ defined then we rewrite using $d \gg a \odot x \mid d' \gg a' \odot y \hookrightarrow (d \wedge d') \gg (a\gamma a') \odot (x \parallel y)$;
    * If $a\gamma a'$ undefined then we rewrite using $d \gg a \odot x \mid d' \gg a' \odot y \hookrightarrow \delta$;
  - $s \in d \gg a \odot N' \mid d' \gg c \rhd N'$, which rewrites using $d \gg a \odot x \mid d' \gg c \rhd y \hookrightarrow \delta$;
  - $s \in d \gg a \odot N' \mid (N' \oplus N')$, which rewrites using $x \mid (y \oplus z) \hookrightarrow x \mid y \oplus x \mid z$.

- $s \in d \gg c \rhd N' \mid N'$, for which we find the cases
  - $s \in d \gg c \rhd N' \mid \delta$, which rewrites using $x \mid \delta \hookrightarrow \delta$;
  - $s \in d \gg c \rhd N' \mid \epsilon$, which rewrites using $d \gg c \rhd x \mid \epsilon \hookrightarrow d \gg c \rhd x$;
  - $s \in d \gg c \rhd N' \mid a$, which rewrites using $d \gg c \rhd x \mid a \hookrightarrow \delta$;
  - $s \in d \gg c \rhd N' \mid c'$, which rewrites using $d \gg c \rhd x \mid c' \hookrightarrow ((d \sim c_{jmp}) \wedge c'_{jmp}) \gg (c \wedge c') \rhd (x \parallel c')$;
  - $s \in d \gg c \rhd N' \mid a \odot N'$, which rewrites using $d \gg c \rhd x \mid a \odot x \hookrightarrow \delta$;

  - $s \in d \gg c \rhd N' \mid c' \rhd N'$, which rewrites using $d \gg c \rhd x \mid c' \rhd y \hookrightarrow ((d \sim c_{jmp}) \wedge c'_{jmp}) \gg (c \wedge c') \rhd \begin{pmatrix} x \mathbin{\lfloor\!\lfloor} c' \blacktriangleright y \oplus \\ y \mathbin{\lfloor\!\lfloor} c \blacktriangleright x \oplus \\ x \mid c' \blacktriangleright y \oplus \\ y \mid c \blacktriangleright x \end{pmatrix}$;
  - $s \in d \gg c \rhd N' \mid d' \gg a$, which rewrites using $d \gg c \rhd x \mid d' \gg a \hookrightarrow \delta$;
  - $s \in d \gg c \rhd N' \mid d' \gg c'$, which rewrites using $d \gg c \rhd x \mid d' \gg c' \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (x \parallel c')$;
  - $s \in d \gg c \rhd N' \mid d' \gg \epsilon$, which rewrites using $d \gg c \rhd x \mid d' \gg \epsilon \hookrightarrow d'^{?} \sim d \gg c \rhd x$;
  - $s \in d \gg c \rhd N' \mid d' \gg a \odot N'$, which rewrites using $d \gg c \rhd x \mid d' \gg a \odot y \hookrightarrow \delta$;
  - $s \in d \gg c \rhd N' \mid d' \gg c' \rhd N'$, which rewrites using $d \gg c \rhd x \mid d' \gg c' \rhd y \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd \begin{pmatrix} x \mathbin{\lfloor\!\lfloor} c' \blacktriangleright y \oplus \\ y \mathbin{\lfloor\!\lfloor} c \blacktriangleright x \oplus \\ x \mid c' \blacktriangleright y \oplus \\ y \mid c \blacktriangleright x \end{pmatrix}$;
  - $s \in d \gg c \rhd N' \mid (N' \oplus N')$, which rewrites using $x \mid (y \oplus z) \hookrightarrow x \mid y \oplus x \mid z$.

- $s \in (N' \oplus N') \mid N'$, which rewrites using $(x \oplus y) \mid z \hookrightarrow x \mid z \oplus y \mid z$.

C.2.9 The form: $s \in \partial_H(N')$

We find the following cases

- $s \in \partial_H(\delta)$, which rewrites using $\partial_H(\delta) \hookrightarrow \delta$;
- $s \in \partial_H(\epsilon)$, which rewrites using $\partial_H(\epsilon) \hookrightarrow \epsilon$;
- $s \in \partial_H(a)$, for which we find the following cases
  - $a \in H$, in which case we have the rule $\partial_H(a) \hookrightarrow \delta$
  - $a \notin H$, in which case we have the rule $\partial_H(a) \hookrightarrow a$
- $s \in \partial_H(c)$, which rewrites using $\partial_H(c) \hookrightarrow c$;
- $s \in \partial_H(a \odot N')$, which rewrites using $\partial_H(x \odot y) \hookrightarrow \partial_H(x) \odot \partial_H(y)$;
- $s \in \partial_H(c \rhd N')$, which rewrites using $\partial_H(x \rhd y) \hookrightarrow \partial_H(x) \rhd \partial_H(y)$;
- $s \in \partial_H(d \gg a)$, which rewrites using $\partial_H(d \gg x) \hookrightarrow d \gg \partial_H(x)$;
- $s \in \partial_H(d \gg c)$, which rewrites using $\partial_H(d \gg x) \hookrightarrow d \gg \partial_H(x)$;
- $s \in \partial_H(d \gg \epsilon)$, which rewrites using $\partial_H(d \gg x) \hookrightarrow d \gg \partial_H(x)$;

- $s \in \partial_H\,(d \gg a \odot N')$, which rewrites using $\partial_H\,(d \gg x) \hookrightarrow d \gg \partial_H\,(x)$;
- $s \in \partial_H\,(d \gg c \rhd N')$, which rewrites using $\partial_H\,(d \gg x) \hookrightarrow d \gg \partial_H\,(x)$;
- $s \in \partial_H\,(N' \oplus N')$, which rewrites using $\partial_H\,(x \oplus y) \hookrightarrow \partial_H\,(x) \oplus \partial_H\,(y)$.

C.3 The rewrite system is strongly normalizing

That the above rewrite system is strongly normalizing can be demonstrated using semantical labelling in combination with the recursive path ordering technique as (among others) described in [7]. We define the following ranking-norm on HyPA terms and reinitialization-clauses.

- $\lfloor \delta \rfloor = \lfloor \epsilon \rfloor = \lfloor a \rfloor = \lfloor c \rfloor = 1$;
- $\lfloor d \gg x \rfloor = \lfloor x \rfloor + 1$;
- $\lfloor x \oplus y \rfloor = \max(\lfloor x \rfloor, \lfloor y \rfloor)$;
- $\lfloor x \odot y \rfloor = \lfloor x \blacktriangleright y \rfloor = \lfloor x \rhd y \rfloor = 2\lfloor x \rfloor + \lfloor y \rfloor$;
- $\lfloor x \,\|\, y \rfloor = \lfloor x \,\lfloor\!\lfloor\, y \rfloor = \lfloor x \,|\, y \rfloor = \lfloor x \rfloor + \lfloor y \rfloor$;
- $\lfloor \partial_H\,(x) \rfloor = \lfloor x \rfloor + 1$.

Now, we label every operator in a HyPA term with its norm. I.e. we write $x \odot_{2\lfloor x \rfloor + \lfloor y \rfloor} y$ in stead of $x \odot y$. Then, we define the following (well-founded) ordering on labelled operators. (Note that we still treat $d \gg x$ as a unary operator.)

- $\delta_1 < \epsilon_1 < a_1 < c_1 < \oplus_1$
- for all n we have $\oplus_n < \oplus_{n+1}$;
- for all n, m, d we have $\oplus_n < d \gg_m$;
- for all n, d, d' we have $d \gg_n < d' \gg_{n+1}$;
- for all n, m, d we have $d \gg_n < \odot_m$;
- for all n we have $\odot_n < \rhd_n$;
- for all n we have $\rhd_n < \blacktriangleright_n$;
- for all n we have $\blacktriangleright_n < \odot_{n+1}$;
- for all n, m we have $\blacktriangleright_n < |_m$;
- for all n we have $|_n < \lfloor\!\lfloor_n$;
- for all n we have $\lfloor\!\lfloor_n < \|_n$;
- for all n we have $\|_n < |_{n+1}$;

- for all n, m we have $\|_n < \partial_{m,H}()$;

In the remainder, we will show for each of the rules that they are strictly decreasing with respect to the recursive path ordering based on $>$ (which we denote $>_{rpo}$). For reasons of readability, we will sometimes omit the labelling of some of the operators, if the exact labelling is not important for the proof. The rules that result in deadlock, or other constants, are considered trivial. Note that for every term x we have $\lfloor x \rfloor \geq 1$.

C.3.1 $d \gg d' \gg x \hookrightarrow (d \sim d') \gg x$

$d \gg_{\lfloor x \rfloor+2} d' \gg_{\lfloor x \rfloor+1} x$
$>_{rpo} \quad d \gg^*_{\lfloor x \rfloor+2} d' \gg_{\lfloor x \rfloor+1} x$
$>_{rpo} \quad (d \sim d') \gg_{\lfloor x \rfloor+1} d \gg^*_{\lfloor x \rfloor+2} d' \gg_{\lfloor x \rfloor+1} x$
$>_{rpo} \quad (d \sim d') \gg_{\lfloor x \rfloor+1} d' \gg_{\lfloor x \rfloor+1} x$
$>_{rpo} \quad (d \sim d') \gg_{\lfloor x \rfloor+1} d' \gg^*_{\lfloor x \rfloor+1} x$
$>_{rpo} \quad (d \sim d') \gg_{\lfloor x \rfloor+1} x$

C.3.2 $d \gg (x \oplus y) \hookrightarrow d \gg x \oplus d \gg y$

$d \gg_{\lfloor x \rfloor+\lfloor y \rfloor+1} (x \oplus y)$
$>_{rpo} \quad d \gg^*_{\lfloor x \rfloor+\lfloor y \rfloor+1} (x \oplus y)$
$>_{rpo} \quad (d \gg^*_{\lfloor x \rfloor+\lfloor y \rfloor+1} (x \oplus y)) \oplus (d \gg^*_{\lfloor x \rfloor+\lfloor y \rfloor+1} (x \oplus y))$
$>_{rpo} \quad (d \gg_{\lfloor x \rfloor+\lfloor y \rfloor+1} (x \oplus {}^*y)) \oplus (d \gg_{\lfloor x \rfloor+\lfloor y \rfloor+1} (x \oplus {}^*y))$
$>_{rpo} \quad d \gg_{\lfloor x \rfloor+\lfloor y \rfloor+1} x \oplus d \gg_{\lfloor x \rfloor+\lfloor y \rfloor+1} y$
$>_{rpo} \quad d \gg_{\lfloor x \rfloor+1} x \oplus d \gg_{\lfloor y \rfloor+1} y$

C.3.3 $(x \odot y) \odot z \hookrightarrow x \odot (y \odot z)$

$(x \odot_{2\lfloor x \rfloor+\lfloor y \rfloor} y) \odot_{4\lfloor x \rfloor+2\lfloor y \rfloor+\lfloor z \rfloor} z$
$>_{rpo} \quad (x \odot_{2\lfloor x \rfloor+\lfloor y \rfloor} y) \odot_{4\lfloor x \rfloor+2\lfloor y \rfloor+\lfloor z \rfloor} {}^*z$
$>_{rpo} \quad ((x \odot_{2\lfloor x \rfloor+\lfloor y \rfloor} y) \odot_{4\lfloor x \rfloor+2\lfloor y \rfloor)+\lfloor z \rfloor} {}^*z) \odot_{2\lfloor x \rfloor+2\lfloor y \rfloor+\lfloor z \rfloor}$
$((x \odot_{2\lfloor x \rfloor+\lfloor y \rfloor} y) \odot_{4\lfloor x \rfloor+2\lfloor y \rfloor+\lfloor z \rfloor} {}^*z)$
$>_{rpo} \quad (x \odot_{2\lfloor x \rfloor+y} y) \odot_{2\lfloor x \rfloor+2\lfloor y \rfloor+\lfloor z \rfloor}$
$((x \odot_{2\lfloor x \rfloor+\lfloor y \rfloor} y) \odot_{4\lfloor x \rfloor+2\lfloor y \rfloor+\lfloor z \rfloor} {}^*z)$
$>_{rpo} \quad (x \odot_{2\lfloor x \rfloor+\lfloor y \rfloor} {}^*y) \odot_{2\lfloor x \rfloor+2\lfloor y \rfloor+\lfloor z \rfloor}$
$((x \odot_{2\lfloor x \rfloor+\lfloor y \rfloor} y) \odot_{4\lfloor x \rfloor+2\lfloor y \rfloor+\lfloor z \rfloor} {}^*z)$
$>_{rpo} \quad x \odot_{2\lfloor x \rfloor+2\lfloor y \rfloor+\lfloor z \rfloor} ((x \odot_{2\lfloor x \rfloor+\lfloor y \rfloor} y) \odot_{4\lfloor x \rfloor+2\lfloor y \rfloor+\lfloor z \rfloor} {}^*z)$
$>_{rpo} \quad x \odot_{2\lfloor x \rfloor+2\lfloor y \rfloor+\lfloor z \rfloor} ((x \odot_{2\lfloor x \rfloor+\lfloor y \rfloor} {}^*y) \odot_{4\lfloor x \rfloor+2\lfloor y \rfloor+\lfloor z \rfloor} z)$
$>_{rpo} \quad x \odot_{2\lfloor x \rfloor+2\lfloor y \rfloor+\lfloor z \rfloor} (y \odot_{4\lfloor x \rfloor+2\lfloor y \rfloor+\lfloor z \rfloor} z)$
$>_{rpo} \quad x \odot_{2\lfloor x \rfloor+2\lfloor y \rfloor+\lfloor z \rfloor} (y \odot_{2\lfloor y \rfloor+\lfloor z \rfloor} z)$

C.3.4 $(x \oplus y) \odot z \hookrightarrow x \odot z \oplus y \odot z$

$(x \oplus y) \odot z$
$>_{rpo} \ (x \oplus y) \odot {}^{*}z$
$>_{rpo} \ ((x \oplus y) \odot {}^{*}z) \oplus ((x \oplus y) \odot {}^{*}z)$
$>_{rpo} \ ((x \oplus {}^{*}y) \odot z) \oplus ((x \oplus {}^{*}y) \odot z)$
$>_{rpo} \ x \odot z \oplus y \odot z$

C.3.5 $x \blacktriangleright y \hookrightarrow x \rhd y \oplus y$

$x \blacktriangleright_{2\lfloor x \rfloor + \lfloor y \rfloor} y$
$>_{rpo} \ x \blacktriangleright_{2\lfloor x \rfloor + \lfloor y \rfloor} {}^{*}y$
$>_{rpo} \ x \blacktriangleright_{2\lfloor x \rfloor + \lfloor y \rfloor} {}^{*}y \oplus x \blacktriangleright_{2\lfloor x \rfloor + \lfloor y \rfloor} {}^{*}y$
$>_{rpo} \ x \blacktriangleright_{2\lfloor x \rfloor + \lfloor y \rfloor} {}^{*}y \oplus y$
$>_{rpo} \ (x \blacktriangleright_{2\lfloor x \rfloor + \lfloor y \rfloor} {}^{*}y) \rhd_{2\lfloor x \rfloor + \lfloor y \rfloor} (x \blacktriangleright_{2\lfloor x \rfloor + \lfloor y \rfloor} {}^{*}y) \oplus y$
$>_{rpo} \ x \rhd_{2\lfloor x \rfloor + \lfloor y \rfloor} (x \blacktriangleright_{2\lfloor x \rfloor + \lfloor y \rfloor} {}^{*}y) \oplus y$
$>_{rpo} \ x \rhd_{2\lfloor x \rfloor + \lfloor y \rfloor} y \oplus y$

C.3.6 $(c \rhd x) \rhd y \hookrightarrow c \rhd (x \blacktriangleright y)$

$(c \rhd_{2+\lfloor x \rfloor} x) \rhd_{4+2\lfloor x \rfloor + \lfloor y \rfloor} y$
$>_{rpo} \ (c \rhd_{2+x} x) \rhd_{4+2\lfloor x \rfloor + \lfloor y \rfloor} {}^{*}y$
$>_{rpo} \ ((c \rhd_{2+x} x) \rhd_{4+2\lfloor x \rfloor + \lfloor y \rfloor} {}^{*}y) \rhd_{2+2\lfloor x \rfloor + \lfloor y \rfloor}$
$((c \rhd_{2+x} x) \rhd_{4+2\lfloor x \rfloor + \lfloor y \rfloor} {}^{*}y)$
$>_{rpo} \ c \rhd_{2+2\lfloor x \rfloor + \lfloor y \rfloor} ((c \rhd_{2+x} x) \rhd_{4+2\lfloor x \rfloor + \lfloor y \rfloor} {}^{*}y)$
$>_{rpo} \ c \rhd_{2+2\lfloor x \rfloor + \lfloor y \rfloor} ((c \rhd_{2+x} {}^{*}x) \rhd_{4+2\lfloor x \rfloor + \lfloor y \rfloor} y)$
$>_{rpo} \ c \rhd_{2+2\lfloor x \rfloor + \lfloor y \rfloor} (x \rhd_{4+2\lfloor x \rfloor + \lfloor y \rfloor} y)$
$>_{rpo} \ c \rhd_{2+2\lfloor x \rfloor + \lfloor y \rfloor} (x \blacktriangleright_{2\lfloor x \rfloor + \lfloor y \rfloor} y)$

C.3.7 $(x \oplus y) \rhd z \hookrightarrow x \rhd z \oplus y \rhd z$

Similar to the proof of $(x \oplus y) \odot z \hookrightarrow x \odot z \oplus y \odot z$.

C.3.8 $x \,||\, y \hookrightarrow x \lfloor\!\lfloor y \oplus y \lfloor\!\lfloor x \oplus x \,|\, y$

Similar to the proof of $x \blacktriangleright y \hookrightarrow x \rhd y \oplus y$.

C.3.9 $(x \oplus y) \lfloor\!\lfloor z \hookrightarrow x \lfloor\!\lfloor z \oplus y \lfloor\!\lfloor z$

Similar to the proof of $(x \oplus y) \odot z \hookrightarrow x \odot z \oplus y \odot z$, since the semantical labelling is irrelevant for this proof.

C.3.10 $(d \gg \epsilon) \odot x \hookrightarrow d^{?} \gg x$

$(d \gg \epsilon) \odot x$
$>_{rpo} \ (d \gg \epsilon) \odot {}^{*}x$

$$\begin{aligned}
&>_{rpo} \quad d^? \gg (d \gg \epsilon) \odot {}^*x \\
&>_{rpo} \quad d^? \gg x
\end{aligned}$$

C.3.11 $(d \gg x) \mathbin{|\!\lfloor} y \hookrightarrow d \gg x \mathbin{|\!\lfloor} y$

$$\begin{aligned}
&(d \gg x) \mathbin{|\!\lfloor} y \\
&>_{rpo} \quad (d \gg x) \mathbin{|\!\lfloor} {}^*y \\
&>_{rpo} \quad d \gg (d \gg x) \mathbin{|\!\lfloor} {}^*y \\
&>_{rpo} \quad d \gg (d \gg^* x) \mathbin{|\!\lfloor} y \\
&>_{rpo} \quad d \gg x \mathbin{|\!\lfloor} y
\end{aligned}$$

C.3.12 $(a \odot x) \mathbin{|\!\lfloor} y \hookrightarrow a \odot (x \parallel y)$

$$\begin{aligned}
&(a \odot x) \mathbin{|\!\lfloor} y \\
&>_{rpo} \quad (a \odot x) \mathbin{|\!\lfloor} {}^*y \\
&>_{rpo} \quad ((a \odot x) \mathbin{|\!\lfloor} {}^*y) \odot ((a \odot x) \mathbin{|\!\lfloor} {}^*y) \\
&>_{rpo} \quad ((a \odot x) \mathbin{|\!\lfloor} {}^*y) \odot ((a \odot {}^*x) \mathbin{|\!\lfloor} y) \\
&>_{rpo} \quad ((a \odot x) \mathbin{|\!\lfloor} {}^*y) \odot (x \mathbin{|\!\lfloor} y) \\
&>_{rpo} \quad a \odot (x \parallel y)
\end{aligned}$$

C.3.13 $a \mathbin{|\!\lfloor} x \hookrightarrow a \odot x$

Trivial, since $\mathbin{|\!\lfloor}_n \geq \odot_n$.

C.3.14 $(d \gg a) \odot x \hookrightarrow d \gg a \odot x$

Similar to the proof of $(d \gg x) \mathbin{|\!\lfloor} y \hookrightarrow d \gg x \mathbin{|\!\lfloor} y$.

C.3.15 $(d \gg a \odot x) \odot y \hookrightarrow d \gg a \odot (x \odot y)$

For the proof of this rule, we use the fact that $(x \odot y) \odot z \hookrightarrow x \odot (y \odot z)$ is an rpo-decreasing rewrite rule.

$$\begin{aligned}
&(d \gg_{3+\lfloor x \rfloor} a \odot_{2+\lfloor x \rfloor} x) \odot_{6+2\lfloor x \rfloor+\lfloor y \rfloor} y \\
&>_{rpo} \quad (d \gg_{3+\lfloor x \rfloor} a \odot_{2+\lfloor x \rfloor} x) \odot_{6+2\lfloor x \rfloor+\lfloor y \rfloor} {}^*y \\
&>_{rpo} \quad d \gg_{3+2\lfloor x \rfloor+\lfloor y \rfloor} (d \gg_{3+\lfloor x \rfloor} a \odot_{2+\lfloor x \rfloor} x) \odot_{6+2\lfloor x \rfloor+\lfloor y \rfloor} {}^*y \\
&>_{rpo} \quad d \gg_{3+2\lfloor x \rfloor+\lfloor y \rfloor} (d \gg^*_{3+\lfloor x \rfloor} a \odot_{2+\lfloor x \rfloor} x) \odot_{6+2\lfloor x \rfloor+\lfloor y \rfloor} y \\
&>_{rpo} \quad d \gg_{3+2\lfloor x \rfloor+\lfloor y \rfloor} (a \odot_{2+\lfloor x \rfloor} x) \odot_{6+2\lfloor x \rfloor+\lfloor y \rfloor} y \\
&>_{rpo} \quad d \gg_{3+2\lfloor x \rfloor+\lfloor y \rfloor} (a \odot_{2+\lfloor x \rfloor} x) \odot_{4+2\lfloor x \rfloor+\lfloor y \rfloor} y \\
&\hookrightarrow \quad d \gg_{3+2\lfloor x \rfloor+\lfloor y \rfloor} a \odot_{2+2\lfloor x \rfloor+\lfloor y \rfloor} (x \odot_{2\lfloor x \rfloor+\lfloor y \rfloor} y)
\end{aligned}$$

C.3.16 $(a \odot x) \rhd y \hookrightarrow a \odot (x \blacktriangleright y)$

$$\begin{aligned}
&(a \odot_{2+\lfloor x \rfloor} x) \rhd_{4+2\lfloor x \rfloor+\lfloor y \rfloor} y \\
&>_{rpo} \quad (a \odot_{2+\lfloor x \rfloor} x) \rhd_{4+2\lfloor x \rfloor+\lfloor y \rfloor} {}^*y \\
&>_{rpo} \quad ((a \odot_{2+\lfloor x \rfloor} x) \rhd_{4+2\lfloor x \rfloor+\lfloor y \rfloor} {}^*y) \odot_{2+2\lfloor x \rfloor+\lfloor y \rfloor} \\
&((a \odot_{2+\lfloor x \rfloor} x) \rhd_{4+2\lfloor x \rfloor+\lfloor y \rfloor} {}^*y)
\end{aligned}$$

$>_{rpo} \quad a \odot_{2+2\lfloor x\rfloor+\lfloor y\rfloor} ((a \odot_{2+\lfloor x\rfloor} x) \rhd_{4+2\lfloor x\rfloor+\lfloor y\rfloor} {}^*y)$
$>_{rpo} \quad a \odot_{2+2\lfloor x\rfloor+\lfloor y\rfloor} ((a \odot_{2+\lfloor x\rfloor} {}^*x) \rhd_{4+2\lfloor x\rfloor+\lfloor y\rfloor} y)$
$>_{rpo} \quad a \odot_{2+2\lfloor x\rfloor+\lfloor y\rfloor} (x \rhd_{4+2\lfloor x\rfloor+\lfloor y\rfloor} y)$
$>_{rpo} \quad a \odot_{2+2\lfloor x\rfloor+\lfloor y\rfloor} (x \rhd_{2\lfloor x\rfloor+\lfloor y\rfloor} y)$

C.3.17 $a \rhd x \hookrightarrow a \odot (\epsilon \oplus x)$

In this proof, we use that $\lfloor x\rfloor = \max(1, \lfloor x\rfloor) = \lfloor \epsilon \oplus x\rfloor$.
$a \rhd_{2+\lfloor x\rfloor} x$
$>_{rpo} \quad a \rhd_{2+\lfloor x\rfloor} {}^*x$
$>_{rpo} \quad (a \rhd_{2+\lfloor x\rfloor} {}^*x) \odot_{2+\lfloor x\rfloor} (a \rhd_{2+\lfloor x\rfloor} {}^*x)$
$>_{rpo} \quad a \odot_{2+\lfloor x\rfloor} (a \rhd_{2+\lfloor x\rfloor} {}^*x)$
$>_{rpo} \quad a \odot_{2+\lfloor x\rfloor} ((a \rhd_{2+\lfloor x\rfloor} {}^*x) \oplus_{\lfloor x\rfloor} (a \rhd_{2+\lfloor x\rfloor} {}^*x))$
$>_{rpo} \quad a \odot_{2+\lfloor x\rfloor} (\epsilon \oplus_{\lfloor x\rfloor} (a \rhd_{2+\lfloor x\rfloor} {}^*x))$
$>_{rpo} \quad a \odot_{2+\lfloor x\rfloor} (\epsilon \oplus_{\lfloor x\rfloor} x)$

C.3.18 $(d \gg x) \rhd y \hookrightarrow d \gg x \rhd y$

Similar to the proof of $(d \gg x) \parallel\!\!\lfloor\, y \hookrightarrow d \gg x \parallel\!\!\lfloor\, y$.

C.3.19 $(d \gg c \rhd x) \odot y \hookrightarrow d \gg c \rhd (x \odot y)$

$(d \gg_{3+\lfloor x\rfloor} c \rhd_{2+\lfloor x\rfloor} x) \odot_{6+2\lfloor x\rfloor+\lfloor y\rfloor} y$
$>_{rpo} \quad (d \gg_{3+\lfloor x\rfloor} c \rhd_{2+\lfloor x\rfloor} x) \odot_{6+2\lfloor x\rfloor+\lfloor y\rfloor} {}^*y$
$>_{rpo} \quad d \gg_{3+2\lfloor x\rfloor+\lfloor y\rfloor} (d \gg_{3+\lfloor x\rfloor} c \rhd_{2+\lfloor x\rfloor} x) \odot_{6+2\lfloor x\rfloor+\lfloor y\rfloor} {}^*y$
$>_{rpo} \quad d \gg_{3+2\lfloor x\rfloor+\lfloor y\rfloor} ((d \gg_{3+\lfloor x\rfloor} c \rhd_{2+\lfloor x\rfloor} x) \odot_{6+2\lfloor x\rfloor+\lfloor y\rfloor} {}^*y) \rhd_{2+2\lfloor x\rfloor+\lfloor y\rfloor}$
$((d \gg_{3+\lfloor x\rfloor} c \rhd_{2+\lfloor x\rfloor} x) \odot_{6+2\lfloor x\rfloor+\lfloor y\rfloor} {}^*y)$
$>_{rpo} \quad d \gg_{3+2\lfloor x\rfloor+\lfloor y\rfloor} c \rhd_{2+2\lfloor x\rfloor+\lfloor y\rfloor} ((d \gg_{3+\lfloor x\rfloor} c \rhd_{2+\lfloor x\rfloor} x) \odot_{6+2\lfloor x\rfloor+\lfloor y\rfloor} {}^*y)$
$>_{rpo} \quad d \gg_{3+2\lfloor x\rfloor+\lfloor y\rfloor} c \rhd_{2+2\lfloor x\rfloor+\lfloor y\rfloor} ((d \gg^*_{3+\lfloor x\rfloor} c \rhd_{2+\lfloor x\rfloor} x) \odot_{6+2\lfloor x\rfloor+\lfloor y\rfloor} y)$
$>_{rpo} \quad d \gg_{3+2\lfloor x\rfloor+\lfloor y\rfloor} c \rhd_{2+2\lfloor x\rfloor+\lfloor y\rfloor} ((c \rhd_{2+\lfloor x\rfloor} x) \odot_{6+2\lfloor x\rfloor+\lfloor y\rfloor} y)$
$>_{rpo} \quad d \gg_{3+2\lfloor x\rfloor+\lfloor y\rfloor} c \rhd_{2+2\lfloor x\rfloor+\lfloor y\rfloor} ((c \rhd_{2+\lfloor x\rfloor} {}^*x) \odot_{6+2\lfloor x\rfloor+\lfloor y\rfloor} y)$
$>_{rpo} \quad d \gg_{3+2\lfloor x\rfloor+\lfloor y\rfloor} c \rhd_{2+2\lfloor x\rfloor+\lfloor y\rfloor} (x \odot_{6+2\lfloor x\rfloor+\lfloor y\rfloor} y)$
$>_{rpo} \quad d \gg_{3+2\lfloor x\rfloor+\lfloor y\rfloor} c \rhd_{2+2\lfloor x\rfloor+\lfloor y\rfloor} (x \odot_{2\lfloor x\rfloor+\lfloor y\rfloor} y)$

C.3.20 $(c \rhd x) \odot y \hookrightarrow c \rhd (x \odot y)$

$(c \rhd_{2+\lfloor x\rfloor} x) \odot_{4+2\lfloor x\rfloor+\lfloor y\rfloor} y$
$>_{rpo} \quad (c \rhd_{2+\lfloor x\rfloor} x) \odot_{4+2\lfloor x\rfloor+\lfloor y\rfloor} {}^*y$
$>_{rpo} \quad ((c \rhd_{2+\lfloor x\rfloor} x) \odot_{4+2\lfloor x\rfloor+\lfloor y\rfloor} {}^*y) \rhd_{2+2\lfloor x\rfloor+\lfloor y\rfloor}$
$((c \rhd_{2+\lfloor x\rfloor} x) \odot_{4+2\lfloor x\rfloor+\lfloor y\rfloor} {}^*y)$
$>_{rpo} \quad c \rhd_{2+2\lfloor x\rfloor+\lfloor y\rfloor} ((c \rhd_{2+\lfloor x\rfloor} x) \odot_{4+2\lfloor x\rfloor+\lfloor y\rfloor} {}^*y)$
$>_{rpo} \quad c \rhd_{2+2\lfloor x\rfloor+\lfloor y\rfloor} ((c \rhd_{2+\lfloor x\rfloor} {}^*x) \odot_{4+2\lfloor x\rfloor+\lfloor y\rfloor} y)$
$>_{rpo} \quad c \rhd_{2+2\lfloor x\rfloor+\lfloor y\rfloor} (x \odot_{4+2\lfloor x\rfloor+\lfloor y\rfloor} y)$
$>_{rpo} \quad c \rhd_{2+2\lfloor x\rfloor+\lfloor y\rfloor} (x \odot_{2\lfloor x\rfloor+\lfloor y\rfloor} y)$

C.3.21 $(d \gg c) \odot x \hookrightarrow d \gg c$

$(d \gg c) \odot x >_{rpo} (d \gg c) \odot {}^{*}x >_{rpo} d \gg c$

C.3.22 $(x \oplus y) \,|\, z \hookrightarrow x \,|\, z \oplus y \,|\, z$

Similar to the proof of $(x \oplus y) \odot z \hookrightarrow x \odot z \oplus y \odot z$, since the semantical labelling is irrelevant for this proof.

C.3.23 $x \,|\, (y \oplus z) \hookrightarrow x \,|\, y \oplus x \,|\, z$

$x \,|\, (y \oplus z)$

$>_{rpo} \quad x \,|\, {}^{*}(y \oplus z) \;\; >_{rpo} \quad (x \,|\, {}^{*}(y \oplus z)) \oplus (x \,|\, {}^{*}(y \oplus z))$

$>_{rpo} \quad (x \,|\, (y \oplus {}^{*}z)) \oplus (x \,|\, (y \oplus {}^{*}z))$

$>_{rpo} \quad (x \,|\, y) \oplus (x \,|\, z)$

C.3.24 $d \gg \epsilon \,|\, d' \gg \epsilon \hookrightarrow (d^{?} \wedge d'^{?}) \gg \epsilon$

$d \gg \epsilon \,|\, d' \gg \epsilon$

$>_{rpo} \quad d \gg \epsilon \,|\, {}^{*}d' \gg \epsilon$

$>_{rpo} \quad (d^{?} \wedge d'^{?}) \gg (d \gg \epsilon \,|\, {}^{*}d' \gg \epsilon)$

$>_{rpo} \quad (d^{?} \wedge d'^{?}) \gg \epsilon$

C.3.25 $d \gg \epsilon \,|\, \epsilon \hookrightarrow d^{?} \gg \epsilon$

Similar to the proof of $d \gg \epsilon \,|\, d' \gg \epsilon \hookrightarrow (d^{?} \wedge d'^{?}) \gg \epsilon$.

C.3.26 $\epsilon \,|\, d \gg \epsilon \hookrightarrow d^{?} \gg \epsilon$

Similar to the proof of $d \gg \epsilon \,|\, d' \gg \epsilon \hookrightarrow (d^{?} \wedge d'^{?}) \gg \epsilon$.

C.3.27 $d \gg a \odot x \,|\, d' \gg a' \odot y \hookrightarrow (d \wedge d') \gg (a \gamma a') \odot (x \,\|\, y)$

$d \gg a \odot x \,|_{6+\lfloor x \rfloor+\lfloor y \rfloor} d' \gg a' \odot y$

$>_{rpo} \quad d \gg a \odot x \,|_{6+\lfloor x \rfloor+\lfloor y \rfloor} {}^{*}d' \gg a' \odot y$

$>_{rpo} \quad (d \wedge d') \gg (d \gg a \odot x \,|_{6+\lfloor x \rfloor+\lfloor y \rfloor} {}^{*}d' \gg a' \odot y)$

$>_{rpo} \quad (d \wedge d') \gg (d \gg a \odot x \,|_{6+\lfloor x \rfloor+\lfloor y \rfloor} {}^{*}d' \gg a' \odot y) \odot$
$(d \gg a \odot x \,|_{6+\lfloor x \rfloor+\lfloor y \rfloor} {}^{*}d' \gg a' \odot y)$

$>_{rpo} \quad (d \wedge d') \gg (a \gamma a') \odot (d \gg a \odot x \,|_{6+\lfloor x \rfloor+\lfloor y \rfloor} {}^{*}d' \gg a' \odot y)$

$>_{rpo} \quad (d \wedge d') \gg (a \gamma a') \odot (d \gg^{*} a \odot x \,|_{6+\lfloor x \rfloor+\lfloor y \rfloor} d' \gg^{*} a' \odot y)$

$>_{rpo} \quad (d \wedge d') \gg (a \gamma a') \odot (a \odot x \,|_{6+\lfloor x \rfloor+\lfloor y \rfloor} a' \odot y)$

$>_{rpo} \quad (d \wedge d') \gg (a \gamma a') \odot (a \odot {}^{*}x \,|_{6+\lfloor x \rfloor+\lfloor y \rfloor} a' \odot {}^{*}y)$

$>_{rpo} \quad (d \wedge d') \gg (a \gamma a') \odot (x \,|_{6+\lfloor x \rfloor+\lfloor y \rfloor} y)$

$>_{rpo} \quad (d \wedge d') \gg (a \gamma a') \odot (x \,\|_{\lfloor x \rfloor+\lfloor y \rfloor} y)$

C.3.28 $a \odot x \,|\, d \gg a' \odot y \hookrightarrow ([\,true\,] \wedge d) \gg (a\gamma a') \odot (x \,\|\, y)$

Similar to the proof of $d \gg a \odot x \,|\, d' \gg a' \odot y \hookrightarrow (d \wedge d') \gg (a\gamma a') \odot (x \,\|\, y)$.

C.3.29 $a \odot x \,|\, a' \odot y \hookrightarrow (a\gamma a') \odot (x \,\|\, y)$

Similar to the proof of $d \gg a \odot x \,|\, d' \gg a' \odot y \hookrightarrow (d \wedge d') \gg (a\gamma a') \odot (x \,\|\, y)$, but leaving out the introduction of $(d \wedge d') \gg$.

C.3.30 $d \gg a \odot x \,|\, a' \odot y \hookrightarrow (d \wedge [\,true\,]) \gg (a\gamma a') \odot (x \,\|\, y)$

Similar to the proof of $d \gg a \odot x \,|\, d' \gg a' \odot y \hookrightarrow (d \wedge d') \gg (a\gamma a') \odot (x \,\|\, y)$.

C.3.31 $d \gg a \,|\, d' \gg a' \odot x \hookrightarrow (d \wedge d') \gg (a\gamma a') \odot x$

$d \gg a \,|\, d' \gg a' \odot x$
$>_{rpo} \quad d \gg a \,|^{*} d' \gg a' \odot x$
$>_{rpo} \quad (d \wedge d') \gg (d \gg a \,|^{*} d' \gg a' \odot x)$
$>_{rpo} \quad (d \wedge d') \gg (d \gg a \,|^{*} d' \gg a' \odot x) \odot$
$(d \gg a \,|^{*} d' \gg a' \odot x)$
$>_{rpo} \quad (d \wedge d') \gg (a\gamma a') \odot (d \gg a \,|^{*} d' \gg a' \odot x)$
$>_{rpo} \quad (d \wedge d') \gg (a\gamma a') \odot x$

C.3.32 $d \gg a \odot x \,|\, d' \gg a' \hookrightarrow (d \wedge d') \gg (a\gamma a') \odot x$

Similar to the proof of $d \gg a \,|\, d' \gg a' \odot x \hookrightarrow (d \wedge d') \gg (a\gamma a') \odot x$.

C.3.33 $d \gg a \odot x \,|\, a' \hookrightarrow (d \wedge [\,true\,]) \gg (a\gamma a') \odot x$

Similar to the proof of $d \gg a \,|\, d' \gg a' \odot x \hookrightarrow (d \wedge d') \gg (a\gamma a') \odot x$.

C.3.34 $a \,|\, d \gg a' \odot x \hookrightarrow ([\,true\,] \wedge d) \gg (a\gamma a') \odot x$

Similar to the proof of $d \gg a \,|\, d' \gg a' \odot x \hookrightarrow (d \wedge d') \gg (a\gamma a') \odot x$.

C.3.35 $a \odot x \,|\, d \gg a' \hookrightarrow ([\,true\,] \wedge d) \gg (a\gamma a') \odot x$

Similar to the proof of $d \gg a \,|\, d' \gg a' \odot x \hookrightarrow (d \wedge d') \gg (a\gamma a') \odot x$.

C.3.36 $d \gg a \,|\, a' \odot x \hookrightarrow (d \wedge [\,true\,]) \gg (a\gamma a') \odot x$

Similar to the proof of $d \gg a \,|\, d' \gg a' \odot x \hookrightarrow (d \wedge d') \gg (a\gamma a') \odot x$.

C.3.37 $d \gg a \,|\, d' \gg a' \hookrightarrow (d \wedge d') \gg (a\gamma a')$

$d \gg a \,|\, d' \gg a'$
$>_{rpo} \quad d \gg a \,|^{*} d' \gg a'$
$>_{rpo} \quad (d \wedge d') \gg (d \gg a \,|^{*} d' \gg a')$
$>_{rpo} \quad (d \wedge d') \gg (a\gamma a')$

C.3.38 $d \gg a \,|\, a' \hookrightarrow (d \wedge [\,true\,]) \gg (a\gamma a')$

Similar to the proof of $d \gg a \,|\, d' \gg a' \hookrightarrow (d \wedge d') \gg (a\gamma a')$.

C.3.39 $a \,|\, d \gg a' \hookrightarrow ([\,true\,] \wedge d) \gg (a\gamma a')$

Similar to the proof of $d \gg a \,|\, d' \gg a' \hookrightarrow (d \wedge d') \gg (a\gamma a')$.

C.3.40 $a \,|\, a' \odot x \hookrightarrow (a\gamma a') \odot x$

$a \,|\, a' \odot x$
$>_{rpo}$ $a \,|^* a' \odot x$
$>_{rpo}$ $(a \,|^* a' \odot x) \odot (a \,|^* a' \odot x)$
$>_{rpo}$ $(a\gamma a') \odot (a \,|^* a' \odot x)$
$>_{rpo}$ $(a\gamma a') \odot x$

C.3.41 $a \odot x \,|\, a' \hookrightarrow (a\gamma a') \odot x$

Similar to the proof of $a \,|\, a' \odot x \hookrightarrow (a\gamma a') \odot x$.

C.3.42 $d \gg \epsilon \,|\, d' \gg c \rhd x \hookrightarrow (d^? \sim d') \gg c \rhd x$

$d \gg \epsilon \,|\, d' \gg c \rhd x$
$>_{rpo}$ $d \gg \epsilon \,|^* d' \gg c \rhd x$
$>_{rpo}$ $(d^? \sim d') \gg (d \gg \epsilon \,|^* d' \gg c \rhd x)$
$>_{rpo}$ $(d^? \sim d') \gg (d \gg \epsilon \,|^* d' \gg c \rhd x) \rhd$
$(d \gg \epsilon \,|^* d' \gg c \rhd x)$
$>_{rpo}$ $(d^? \sim d') \gg c \rhd (d \gg \epsilon \,|^* d' \gg c \rhd x)$
$>_{rpo}$ $(d^? \sim d') \gg c \rhd x$

C.3.43 $d \gg c \rhd x \,|\, d' \gg \epsilon \hookrightarrow (d'^? \sim d) \gg c \rhd x$

Similar to the proof of $d \gg \epsilon \,|\, d' \gg c \rhd x \hookrightarrow (d^? \sim d') \gg c \rhd x$.

C.3.44 $c \rhd x \,|\, d \gg \epsilon \hookrightarrow d^? \gg c \rhd x$

Similar to the proof of $d \gg \epsilon \,|\, d' \gg c \rhd x \hookrightarrow (d^? \sim d') \gg c \rhd x$.

C.3.45 $d \gg \epsilon \,|\, c \rhd x \hookrightarrow d^? \gg c \rhd x$

Similar to the proof of $d \gg \epsilon \,|\, d' \gg c \rhd x \hookrightarrow (d^? \sim d') \gg c \rhd x$.

C.3.46 $\epsilon \,|\, d \gg c \rhd x \hookrightarrow d \gg c \rhd x$

Similar to the proof of $d \gg \epsilon \,|\, d' \gg c \rhd x \hookrightarrow (d^? \sim d') \gg c \rhd x$.

C.3.47 $d \gg c \rhd x \,|\, \epsilon \hookrightarrow d \gg c \rhd x$

Similar to the proof of $d \gg \epsilon \,|\, d' \gg c \rhd x \hookrightarrow (d^? \sim d') \gg c \rhd x$.

C.3.48 $d \gg c \,|\, d' \gg \epsilon \hookrightarrow (d'^{?} \sim d) \gg c$

$d \gg c \,|\, d' \gg \epsilon$
$>_{rpo} \quad d \gg c \,|\, {}^{*}d' \gg \epsilon$
$>_{rpo} \quad (d'^{?} \sim d) \gg (d \gg c \,|\, {}^{*}d' \gg \epsilon)$
$>_{rpo} \quad (d'^{?} \sim d) \gg c$

C.3.49 $d \gg \epsilon \,|\, d' \gg c \hookrightarrow (d^{?} \sim d') \gg c$

Similar to the proof of $d \gg c \,|\, d' \gg \epsilon \hookrightarrow (d'^{?} \sim d) \gg c$.

C.3.50 $\epsilon \,|\, c \rhd x \hookrightarrow c \rhd x$

Trivial, since $c \rhd x$ is a subterm of $\epsilon \,|\, c \rhd x$.

C.3.51 $c \rhd x \,|\, \epsilon \hookrightarrow c \rhd x$

Trivial, since $c \rhd x$ is a subterm of $c \rhd x \,|\, \epsilon$.

C.3.52 $c \,|\, d \gg \epsilon \hookrightarrow d^{?} \gg c$

Similar to the proof of $d \gg c \,|\, d' \gg \epsilon \hookrightarrow (d'^{?} \sim d) \gg c$.

C.3.53 $d \gg \epsilon \,|\, c \hookrightarrow d^{?} \gg c$

Similar to the proof of $d \gg c \,|\, d' \gg \epsilon \hookrightarrow (d'^{?} \sim d) \gg c$.

C.3.54 $\epsilon \,|\, d \gg c \hookrightarrow d \gg c$

Similar to the proof of $d \gg c \,|\, d' \gg \epsilon \hookrightarrow (d'^{?} \sim d) \gg c$.

C.3.55 $d \gg c \,|\, \epsilon \hookrightarrow d \gg c$

Similar to the proof of $d \gg c \,|\, d' \gg \epsilon \hookrightarrow (d'^{?} \sim d) \gg c$.

C.3.56 $d \gg c \rhd x \,|\, d' \gg c' \rhd y \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd$
$(x \,\lfloor\!\lfloor\, c' \blacktriangleright y \;\oplus\; y \,\lfloor\!\lfloor\, c \blacktriangleright x \;\oplus\; x \,|\, c' \blacktriangleright y \;\oplus\; y \,|\, c \blacktriangleright x)$

$d \gg c \rhd x \,|_{6+\lfloor x \rfloor+\lfloor y \rfloor}\, d' \gg c' \rhd y$
$>_{rpo} \quad d \gg c \rhd x \,|_{6+\lfloor x \rfloor+\lfloor y \rfloor}\, {}^{*}d' \gg c' \rhd y$
$>_{rpo} \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (d \gg c \rhd x \,|_{6+\lfloor x \rfloor+\lfloor y \rfloor}\, {}^{*}d' \gg c' \rhd y)$
$>_{rpo} \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (d \gg c \rhd x \,|_{6+\lfloor x \rfloor+\lfloor y \rfloor}\, {}^{*}d' \gg c' \rhd y) \rhd$
$(d \gg c \rhd x \,|_{6+\lfloor x \rfloor+\lfloor y \rfloor}\, {}^{*}d' \gg c' \rhd y)$
$>_{rpo} \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd$
$(d \gg c \rhd x \,|_{6+\lfloor x \rfloor+\lfloor y \rfloor}\, {}^{*}d' \gg c' \rhd y)$
$>_{rpo} \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd$

$$\left(\begin{array}{l}(d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \oplus\\ (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \oplus\\ (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \oplus\\ (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y)\end{array}\right)$$

$$>_{rpo} \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd$$

$$\left(\begin{array}{l}(d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \lfloor\!\lfloor_{2+\lfloor x\rfloor+\lfloor y\rfloor} (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \oplus\\ (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \lfloor\!\lfloor_{2+\lfloor x\rfloor+\lfloor y\rfloor} (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \oplus\\ (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y)\,|_{2+\lfloor x\rfloor+\lfloor y\rfloor} (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \oplus\\ (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y)\,|_{2+\lfloor x\rfloor+\lfloor y\rfloor} (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y)\end{array}\right)$$

$$>_{rpo} \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd$$

$$\left(\begin{array}{l}x \lfloor\!\lfloor_{2+\lfloor x\rfloor+\lfloor y\rfloor} (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \oplus\\ y \lfloor\!\lfloor_{2+\lfloor x\rfloor+\lfloor y\rfloor} (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \oplus\\ x\,|_{2+\lfloor x\rfloor+\lfloor y\rfloor} (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \oplus\\ y\,|_{2+\lfloor x\rfloor+\lfloor y\rfloor} (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y)\end{array}\right)$$

$$>_{rpo} \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd$$

$$\left(\begin{array}{l}x \lfloor\!\lfloor_{2+\lfloor x\rfloor+\lfloor y\rfloor} (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \blacktriangleright (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \oplus\\ y \lfloor\!\lfloor_{2+\lfloor x\rfloor+\lfloor y\rfloor} (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \blacktriangleright (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \oplus\\ x\,|_{2+\lfloor x\rfloor+\lfloor y\rfloor} (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \blacktriangleright (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \oplus\\ y\,|_{2+\lfloor x\rfloor+\lfloor y\rfloor} (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \blacktriangleright (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y)\end{array}\right)$$

$$>_{rpo} \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd$$

$$\left(\begin{array}{l}x \lfloor\!\lfloor_{2+\lfloor x\rfloor+\lfloor y\rfloor} c' \blacktriangleright (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \oplus\\ y \lfloor\!\lfloor_{2+\lfloor x\rfloor+\lfloor y\rfloor} c \blacktriangleright (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \oplus\\ x\,|_{2+\lfloor x\rfloor+\lfloor y\rfloor} c' \blacktriangleright (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y) \oplus\\ y\,|_{2+\lfloor x\rfloor+\lfloor y\rfloor} c \blacktriangleright (d \gg c \rhd x\,|_{6+\lfloor x\rfloor+\lfloor y\rfloor}\, {}^*d' \gg c' \rhd y)\end{array}\right)$$

$$>_{rpo} \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd$$

$$\left(\begin{array}{l}x \lfloor\!\lfloor_{2+\lfloor x\rfloor+\lfloor y\rfloor} c' \blacktriangleright y \oplus\\ y \lfloor\!\lfloor_{2+\lfloor x\rfloor+\lfloor y\rfloor} c \blacktriangleright x \oplus\\ x\,|_{2+\lfloor x\rfloor+\lfloor y\rfloor} c' \blacktriangleright y \oplus\\ y\,|_{2+\lfloor x\rfloor+\lfloor y\rfloor} c \blacktriangleright x\end{array}\right)$$

C.3.57 $c \rhd x\,|\,d \gg c' \rhd y \hookrightarrow (c_{jmp} \wedge (d \sim c'_{jmp})) \gg (c \wedge c') \rhd$
$(x \lfloor\!\lfloor c' \blacktriangleright y \;\oplus\; y \lfloor\!\lfloor c \blacktriangleright x \;\oplus\; x\,|\,c' \blacktriangleright y \;\oplus\; y\,|\,c \blacktriangleright x)$

Similar to the proof of $d \gg c \rhd x\,|\,d' \gg c' \rhd y \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (x \lfloor\!\lfloor c' \blacktriangleright y \;\oplus\; y \lfloor\!\lfloor c \blacktriangleright x \;\oplus\; x\,|\,c' \blacktriangleright y \;\oplus\; y\,|\,c \blacktriangleright x)$.

C.3.58 $d \gg c \rhd x\,|\,c' \rhd y \hookrightarrow ((d \sim c_{jmp}) \wedge c'_{jmp}) \gg (c \wedge c') \rhd$
$(x \lfloor\!\lfloor c' \blacktriangleright y \;\oplus\; y \lfloor\!\lfloor c \blacktriangleright x \;\oplus\; x\,|\,c' \blacktriangleright y \;\oplus\; y\,|\,c \blacktriangleright x)$

Similar to the proof of $d \gg c \rhd x\,|\,d' \gg c' \rhd y \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (x \lfloor\!\lfloor c' \blacktriangleright y \;\oplus\; y \lfloor\!\lfloor c \blacktriangleright x \;\oplus\; x\,|\,c' \blacktriangleright y \;\oplus\; y\,|\,c \blacktriangleright x)$.

C.3.59 $c \rhd x \mid c' \rhd y \hookrightarrow (c \wedge c')_{jmp} \gg (c \wedge c') \rhd$
$(x \mathbin{\lfloor\!\lfloor} c' \blacktriangleright y \oplus y \mathbin{\lfloor\!\lfloor} c \blacktriangleright x \oplus x \mid c' \blacktriangleright y \oplus y \mid c \blacktriangleright x)$

Similar to the proof of $d \gg c \rhd x \mid d' \gg c' \rhd y \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (x \mathbin{\lfloor\!\lfloor} c' \blacktriangleright y \oplus y \mathbin{\lfloor\!\lfloor} c \blacktriangleright x \oplus x \mid c' \blacktriangleright y \oplus y \mid c \blacktriangleright x)$, but using the fact that $[\, \mathit{true} \,] \sim c_{jmp}$ is logically equivalent to c.

C.3.60 $d \gg c \mid d' \gg c' \rhd x \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (c \parallel x)$

$d \gg c \,|_{5+\lfloor x \rfloor}\, d' \gg c' \rhd x$
$>_{rpo} \quad d \gg c \,|_{5+\lfloor x \rfloor}\, {}^{*}d' \gg c' \rhd x$
$>_{rpo} \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (d \gg c \,|_{5+\lfloor x \rfloor}\, {}^{*}d' \gg c' \rhd x)$
$>_{rpo} \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (d \gg c \,|_{5+\lfloor x \rfloor}\, {}^{*}d' \gg c' \rhd x) \rhd$
$(d \gg c \,|_{5+\lfloor x \rfloor}\, {}^{*}d' \gg c' \rhd x)$
$>_{rpo} \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (d \gg c \,|_{5+\lfloor x \rfloor}\, {}^{*}d' \gg c' \rhd x)$
$>_{rpo} \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (d \gg^{*} c \,|_{5+\lfloor x \rfloor}\, d' \gg c' \rhd {}^{*}x)$
$>_{rpo} \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (c \,|_{5+\lfloor x \rfloor}\, d' \gg c' \rhd {}^{*}x)$
$>_{rpo} \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (c \,|_{5+\lfloor x \rfloor}\, x)$
$>_{rpo} \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (c \,\|_{1+\lfloor x \rfloor}\, x)$

C.3.61 $d \gg c \rhd x \mid d' \gg c' \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (x \parallel c')$

Similar to the proof of $d \gg c \mid d' \gg c' \rhd x \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (c \parallel x)$.

C.3.62 $d \gg c \rhd x \mid c' \hookrightarrow ((d \sim c_{jmp}) \wedge c'_{jmp}) \gg (c \wedge c') \rhd (x \parallel c')$

Similar to the proof of $d \gg c \mid d' \gg c' \rhd x \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (c \parallel x)$.

C.3.63 $c \rhd x \mid d \gg c' \hookrightarrow (c_{jmp} \wedge (d \sim c'_{jmp})) \gg (c \wedge c') \rhd (x \parallel c')$

Similar to the proof of $d \gg c \mid d' \gg c' \rhd x \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (c \parallel x)$.

C.3.64 $d \gg c \mid c' \rhd x \hookrightarrow ((d \sim c_{jmp}) \wedge c'_{jmp}) \gg (c \wedge c') \rhd (c \parallel x)$

Similar to the proof of $d \gg c \mid d' \gg c' \rhd x \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (c \parallel x)$.

C.3.65 $c \mid d \gg c' \rhd x \hookrightarrow (c_{jmp} \wedge (d \sim c'_{jmp})) \gg (c \wedge c') \rhd (c \parallel x)$

Similar to the proof of $d \gg c \mid d' \gg c' \rhd x \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (c \parallel x)$.

C.3.66 $d \gg c \mid d' \gg c' \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c')$

$d \gg c \mid d' \gg c'$
$>_{rpo} \quad d \gg c \mid {}^{*}d' \gg c'$
$>_{rpo} \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (d \gg c \mid {}^{*}d' \gg c')$
$>_{rpo} \quad ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c')$

C.3.67 $c \mid d \gg c' \hookrightarrow (c_{jmp} \wedge (d \sim c'_{jmp})) \gg (c \wedge c')$

Similar to the proof of $d \gg c \mid d' \gg c' \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c')$.

C.3.68 $d \gg c \mid c' \hookrightarrow ((d \sim c_{jmp}) \wedge c'_{jmp}) \gg (c \wedge c')$

Similar to the proof of $d \gg c \mid d' \gg c' \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c')$.

C.3.69 $c \rhd x \mid c' \hookrightarrow (c \wedge c')_{jmp} \gg (c \wedge c') \rhd (x \parallel c')$

Similar to the proof of $d \gg c \mid d' \gg c' \rhd x \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (c \parallel x)$.

C.3.70 $c \mid c' \rhd x \hookrightarrow (c \wedge c')_{jmp} \gg (c \wedge c') \rhd (c \parallel x)$

Similar to the proof of $d \gg c \mid d' \gg c' \rhd x \hookrightarrow ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg (c \wedge c') \rhd (c \parallel x)$.

C.3.71 $\partial_H (x \odot y) \hookrightarrow \partial_H (x) \odot \partial_H (y)$

$\partial_H (x \odot y)$
$>_{rpo} \quad \partial_H (x \odot y)^*$
$>_{rpo} \quad \partial_H (x \odot y)^* \odot \partial_H (x \odot y)^*$
$>_{rpo} \quad \partial_H (x \odot {}^{*}y) \odot \partial_H (x \odot {}^{*}y)$
$>_{rpo} \quad \partial_H (x) \odot \partial_H (y)$

C.3.72 $\partial_H (x \rhd y) \hookrightarrow \partial_H (x) \rhd \partial_H (y)$

Similar to the proof of $\partial_H (x \odot y) \hookrightarrow \partial_H (x) \odot \partial_H (y)$.

C.3.73 $\partial_H (d \gg x) \hookrightarrow d \gg \partial_H (x)$

$\partial_H (d \gg x)$
$>_{rpo} \quad \partial_H (d \gg x)^*$
$>_{rpo} \quad d \gg \partial_H (d \gg x)^*$
$>_{rpo} \quad d \gg \partial_H (d \gg^{*} x)$
$>_{rpo} \quad d \gg \partial_H (x)$

C.3.74 $\partial_H (x \oplus y) \hookrightarrow \partial_H (x) \oplus \partial_H (y)$

Similar to the proof of $\partial_H (x \odot y) \hookrightarrow \partial_H (x) \odot \partial_H (y)$.