TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY

Linearization of hybrid processes

Citation for published version (APA):
Brand, van den, P. C. W., Reniers, M. A., & Cuijpers, P. J. L. (2004). *Linearization of hybrid processes*. (Computer science reports; Vol. 0429). Technische Universiteit Eindhoven.

Document status and date:
Published: 01/01/2004

Document Version:
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

Download date: 21. Sep. 2023

Linearization of Hybrid Processes

P.C.W. van den Brand M.A. Reniers P.J.L.Cuijpers
Department of Mathematics and Computer Science,
Eindhoven University of Technology (TU/e)

Abstract

We present an algorithm for the linearization of hybrid processes modeled in hybrid process algebra (HyPA) and prove its correctness. HyPA is a formalism that is suitable for the algebraic analysis of hybrid systems, i.e., systems with continuous (physical) as well as discrete (computational) components. Linearization is a useful first step in this analysis, because it reduces the complexity of model descriptions by transforming them into so-called linear form. Furthermore, this linear form allows the use of analysis techniques that cannot be applied to the full HyPA syntax. We also extend HyPA with an abstraction operator.

1 Introduction

A hybrid system is a system with continuous (physical) as well as discrete (computational) components. For instance, a computer program controlling a continuous chemical process is a hybrid system. Both types of components have been studied extensively in isolation, but the interaction between them has only been the subject of research since a few years. This research has led to a number of different formalisms. Among the most well-known formalisms are hybrid automata [10], hybrid I/O automata [11] and hybrid Petri nets [2]. Various process algebra based approaches exist as well, such as Process Algebra for Hybrid Systems [3], hybrid χ [15] and HyPA [7, 5].

Hybrid process algebras attempt to extend the knowledge and experience of the field of process algebra to the field of hybrid systems. This is usually achieved by taking an existing (discrete or timed) process algebra and extending it with elements for modeling continuous behavior. The result is a hybrid formalism that fully supports algebraic reasoning. Moreover, these formalisms provide compositionality of all the operators, including the parallel composition. Of the three hybrid process algebras mentioned above, HyPA was chosen for this work. HyPA is a conservative extension of the discrete process algebra ACP [1], with the disrupt operator from LOTOS [4] and with clauses [17] for the description of continuous behavior and discontinuities.

Hybrid process algebras allow a modeler to construct intuitive models of complex hybrid systems and to reason about these models. However, many analysis techniques take (a description of) a state space as input, but it is far from trivial to generate the state space of a given model. Furthermore, compositionality requires an equivalence that is robust under an arbitrary context. This equivalence is necessarily very strong. Certain analysis techniques, such as the safety analysis of [6], cannot be used under such a strong equivalence.

In this paper, we present an algorithm for the *linearization* of hybrid processes described in the hybrid process algebra HyPA. Linearization in process algebra is a transformation of a (recursive) specification, or model description, into a symbolic representation of the state space. This symbolic representation is expressed as a recursive specification as well, but it uses only a small subset of the full process algebra. Such a specification is said to be linear or in linear form. There are

several advantages to this approach. First, it is fairly straightforward to generate the actual state space of a system if its specification is in linear form. Second, the linear form is convenient for storage and manipulation by tools. Finally, a weaker notion of equivalence can be used on linear specifications, namely the one that is compositional only for the restricted set of operators used in linear specifications.

Linearization was first described in the context of the discrete process algebra μCRL [8]. We used the linearization algorithm of μCRL [9, 16] as a starting point for our own algorithm. μCRL and the process algebraic part of HyPA are very similar, so there are many similarities between the linearization algorithms. However, μCRL does not have any hybrid features, so we adapted the algorithm to incorporate HyPA's hybrid aspects. On the other hand, HyPA does not have an equivalent of μCRL's alternative quantification (denoted $\sum_{d \in D}$ with a possibly infinite domain D). This sum operator is essential for linearizing specifications with large or infinite state spaces, so we introduce an operator for abstraction of model variables that can take over the role of this μCRL operator. Abstraction of model variables is indeed a valuable addition to HyPA in itself.

The structure of this paper is as follows. In section 2, HyPA is presented. In section 3, an abstraction operator for model variables is introduced. In section 4, the linearization algorithm is presented and its correctness is proven. In section 5, some optimizations are discussed and in section 6 an example is given. Finally, in section 7, we conclude with a discussion of the results and possible directions for future research.

2 Hybrid Process Algebra

In this section the syntax and semantics of HyPA are discussed. The discussion presented here is adapted from [6]. A more detailed explanation of HyPA can be found in [7].

2.1 Syntax

The syntax of HyPA is an extension of the process algebra ACP [1], with the disrupt operator from LOTOS [4] and with variants of the flow clauses and re-initialization clauses from the event-flow formalism introduced in [17]. The signature of HyPA consists of the following constant and function symbols:

1. deadlock δ,
2. empty process ϵ,
3. discrete actions $a \in \mathcal{A}$,
4. flow clauses $c \in C$,
5. a family of process re-initialization operators $d \gg \_$ where $d \in D$,
6. alternative composition $\_ \oplus \_$,
7. sequential composition $\_ \odot \_$,
8. disrupt $\_ \blacktriangleright \_$ and left-disrupt $\_ \rhd \_$,
9. parallel composition $\_ \parallel \_$, left-parallel composition $\_ \lfloor\!\lfloor \_$ and forced-synchronization $\_ \mid \_$,
10. a family of encapsulation operators $\partial_H(\_)$ where $H \subseteq \mathcal{A}$.

The binding order of these operators is as follows: $\odot$, $\blacktriangleright$, $\rhd$, $d \gg$, $\parallel$, $\lfloor\!\lfloor$, $\mid$, $\oplus$, where sequential composition binds strongest and alternative composition binds weakest. These constants and operators are described informally below.

Deadlock, empty process and discrete actions The atomic processes δ (called *deadlock*) and ϵ (called *empty process*) are used to model a deadlocking process and a (successfully) terminating process, respectively. The atomic *discrete actions* are used as an abstract model for discrete, computational behavior.

Flow clauses Flow clauses are used to model continuous, never terminating, physical behavior by describing how the model variables $\mathcal{V}_\mathrm{m}$ are allowed to change through time. A flow clause is a pair $(V|\, P_f)$ of a set of model variables $V \subseteq \mathcal{V}_\mathrm{m}$ and a flow predicate $P_f \in \mathcal{P}_f$. The set V models which variables are not allowed to jump at the beginning of a flow.

The predicate P_f models the continuous behavior. More precisely, P_f describes a set of *flows*, where a flow is a (partial) function of time T to the valuations *Val* of model variables. T has a closed-interval domain starting from 0, and *Val* is the set of variable valuations $\mathcal{V}_\mathrm{m} \to \mathcal{V}$, where $\mathcal{V}$ is the union of all variable domains and is defined as $\mathcal{V} = \bigcup_{x \in \mathcal{V}_\mathrm{m}} \mathcal{V}(x)$. The set of all flows is $\mathcal{F} = \{f \in T \mapsto \mathit{Val} \,|\, \mathrm{dom}(f) = [0, t] \text{ for some } t \in T\}$. The flows that are described by a flow predicate are called the solutions of that predicate.

The set of predicates $\mathcal{P}_f$, the sets $\mathcal{V}_\mathrm{m}$ of model variables and T of time points, and the notion of solution $\models_f \subseteq \mathcal{F} \times \mathcal{P}_f$ are parameters of the theory. This means that the modeler can choose an appropriate method to describe flows, for example with differential equations, integrals, algebraic inequalities, etc. Finally, the set of all flow clauses is closed under conjunction ($\wedge$) and it is assumed that there is a flow predicate $\mathit{false} \in \mathcal{P}_f$, which has no solutions.

Re-initializations A process re-initialization $d \gg p$ models the behavior of p where the model variables are submitted to a discontinuous change as specified by the re-initialization clause d. A re-initialization clause describes a set of *re-initializations*, where a re-initialization is a pair of valuations representing the values of the model variables prior to and immediately after the re-initialization. The set of all re-initializations $\mathit{Val} \times \mathit{Val}$ is denoted $\mathcal{R}$.

A re-initialization clause is a pair $[V\,|P_r\,]$ of a set of model variables $V \subseteq \mathcal{V}_\mathrm{m}$ and a re-initialization predicate $P_r \in \mathcal{P}_r$. The set V models which variables are allowed to change. Note that this is precisely opposite to flow clauses, where V denotes those variables that do *not* change (initially). Predicate P_r models the discontinuous changes. In a predicate, x^- denotes the valuation of a variable x before re-initialization, and x^+ denotes the valuation of a variable x after re-initialization. As with flow clauses, the set of predicates $\mathcal{P}_r$ and the notion of solution $\models_r \subseteq \mathcal{R} \times \mathcal{P}_r$ are parameters of the theory.

It is assumed that there is a flow predicate $\mathit{true} \in \mathcal{P}_r$, which satisfies all re-initializations in $\mathcal{R}$, and a predicate $\mathit{false} \in \mathcal{P}_r$, which satisfies no re-initializations in $\mathcal{R}$. The set of all re-initialization clauses is closed under conjunction ($\wedge$), disjunction ($\vee$) and concatenation ($\sim$). Furthermore, there is a satisfiability operator ($d^?$) on re-initialization clauses d, which does not change the valuation of any model variable, but only executes the re-initialized process if d can be satisfied in some way. Finally, there is a re-initialization clause (c_{jmp}) derived from a flow clause c, which executes the same discontinuities that are allowed initially by the flow clause c. These last two operators are mainly used for calculating with process terms.

Alternative and sequential composition The alternative composition $p \oplus q$ models a (non-deterministic) choice between the processes p and q. The sequential composition $p \odot q$ models a sequential execution of processes p and q. The process q is executed after (successful) termination of the process p.

Disrupt and left-disrupt The disrupt $p \blacktriangleright q$ 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 essential for modeling two flow clauses executing one after the other, since the behavior of flow clauses never terminates. The left-disrupt $p \rhd q$ is mainly needed for calculation and axiomatization purposes, rather than for modeling purposes. It first executes a part of the process p and then behaves as a normal disrupt.

Parallel composition The parallel composition $p \parallel q$ 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 of actions (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 (partial, commutative and associative) communication function $\gamma \in \mathcal{A} \times \mathcal{A} \mapsto \mathcal{A}$. For example, if the actions a and a' synchronize, then the resulting action is $a'' = a\gamma a'$. Actions cannot synchronize with flow clauses, and in a parallel composition between those, the action executes first. Re-initializations synchronize only if the processes on which they act synchronize.

As with the left-disrupt, the left-parallel and forced-synchronization operators are mainly introduced for calculation purposes. The left-parallel composition $p \lfloor\!\lfloor q$ denotes that p performs a discrete action first (if possible), and then behaves as a normal parallel composition. The forced-synchronization $p \mid q$ denotes how the first behavior (either a discrete action or a part of a flow) of p and q is synchronized, after which it behaves as a normal parallel composition as well.

Encapsulation Encapsulation $\partial_H(p)$ models that the discrete actions from the set $H \subseteq \mathcal{A}$ are blocked during the execution of process p. This operator is often used in combination with the parallel composition to model that synchronization between discrete actions is enforced.

Recursion Terms can be constructed using variables from a given set of process variables $\mathcal{V}_p$ (with $\mathcal{V}_p \cap \mathcal{V}_m = \emptyset$), as usual. The set of all such terms is denoted $\mathcal{T}(\mathcal{V}_p)$ and these are referred to as terms or open terms. Terms in which no process variables occur are called closed terms. The set of all closed terms is denoted $\mathcal{T}$. Finally, all processes should be interpreted in the light of a set E of recursive definitions of the form $X \approx p$, where X is a process variable and p is a term.

2.2 Semantics

The formal semantics of HyPA is defined in terms of a Hybrid Transition System. Such a transition system has two different kinds of transitions, namely one associated with computational behavior (i.e. discrete actions), and the other associated with physical behavior (i.e. flow clauses).

Definition 1 (Hybrid Transition System) *A* hybrid transition system *is a tuple* $\langle X, A, \Sigma, \mapsto, \rightsquigarrow, \checkmark \rangle$*, consisting of a state space* X*, a set of action labels* A*, a set of flow labels* Σ*, and transition relations* $\mapsto \subseteq X \times A \times X$ *and* $\rightsquigarrow \subseteq X \times \Sigma \times X$*. 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, i.e. $X = \mathcal{T}(\mathcal{V}_p) \times \mathit{Val}$. The set of action labels is formed by pairs of actions and valuations, i.e. $A = \mathcal{A} \times \mathit{Val}$, and the set of flow labels is formed by the set of flows, i.e. $\Sigma = \mathcal{F}$. Recall that the elements $f \in \mathcal{F}$ have a closed-interval domain, possibly a singleton, starting in 0.

The notation $\langle\, x \,\rangle \overset{a}{\mapsto} \langle\, x' \,\rangle$ is used for a transition $(x, a, x') \in \mapsto$ with $x, x' \in X$ and $a \in A$. Similarly, $\langle\, x \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, x' \,\rangle$ is used for a transition $(x, \sigma, x') \in \rightsquigarrow$ with $\sigma \in \Sigma$, and for arbitrary transitions, $\langle\, x \,\rangle \overset{l}{\to} \langle\, x' \,\rangle$ is used instead of $(x, l, x') \in \mapsto \cup \rightsquigarrow$ and $l \in A \cup \mathcal{F}$. Finally, termination is denoted $\langle\, x \,\rangle\checkmark$ instead of $x \in \checkmark$.

First, the definition of a solution of a flow clause and a re-initialization clause is given. Then, the semantics of the HyPA constants and function symbols is given, using deduction rules in the style of [14]. See [7] for a detailed explanation of the semantics.

Definition 2 (Solution of a flow clause) *A pair $(\nu, \sigma) \in Val \times \mathcal{F}$, is defined to be a* solution *of a flow clause $c \in C$, denoted $(\nu, \sigma) \models c$, as follows:*

- $(\nu, \sigma) \models (V|\, \mathcal{P}_f)$ *if* $\sigma \models_f \mathcal{P}_f$, *and for all* $x \in V$ *we find* $\nu(x) = \sigma(0)(x)$;
- $(\nu, \sigma) \models c \wedge c'$ *if* $(\nu, \sigma) \models c$ *and* $(\nu, \sigma) \models c'$.

Definition 3 (Solution of a re-initialization clause) *A re-initialization $(\nu, \nu') \in \mathcal{R}$ is defined to be a* solution *of a re-initialization clause $d \in D$, denoted $(\nu, \nu') \models d$, as follows:*

- $(\nu, \nu') \models [V|\, \mathcal{P}_r]$ *if* $(\nu, \nu') \models_r \mathcal{P}_r$ *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* $\upsilon \in Val$ *with* $(\nu, \upsilon) \models d'$ *and* $(\upsilon, \nu') \models d''$;
- $(\nu, \nu') \models d'^{?}$ *if* $\nu = \nu'$, *and there exists* $\upsilon \in Val$ *with* $(\nu, \upsilon) \models d'$;
- $(\nu, \nu') \models c_{jmp}$ *if there exists* $\sigma \in \Sigma$ *such that* $(\nu, \sigma) \models c$ *and* $\sigma(0) = \nu'$.

In the tables 1, 2 and 3, p, p', q, q' denote process terms, a, a', a'' denote actions, c denotes a flow clause, d denotes a re-initialization clause, H denotes a set of actions, X denotes a recursion variable, ν, ν', ν'' denote valuations, σ denotes a flow, t denotes a point in time, and l denotes an arbitrary transition label.

Table 1: Operational semantics of HyPA

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

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

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

$$\frac{\langle\, p, \nu \,\rangle \overset{l}{\to} \langle\, p', \nu' \,\rangle}{\langle\, p \odot q, \nu \,\rangle \overset{l}{\to} \langle\, p' \odot q, \nu' \,\rangle}(9) \qquad \frac{\langle\, p, \nu \,\rangle\checkmark,\ \langle\, q, \nu \,\rangle \overset{l}{\to} \langle\, q', \nu' \,\rangle}{\langle\, p \odot q, \nu \,\rangle \overset{l}{\to} \langle\, q', \nu' \,\rangle}(10)$$

Table 2: Operational semantics of HyPA, disrupt and left-disrupt

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

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

Table 3: Operational semantics of HyPA, parallel composition, encapsulation and recursion

$$\frac{\langle\, p,\nu\,\rangle\checkmark, \langle\, q,\nu\,\rangle\checkmark}{\begin{array}{l}\langle\, p \parallel q,\nu\,\rangle\checkmark\\ \langle\, p \mid q,\nu\,\rangle\checkmark\end{array}}(15) \qquad \frac{\langle\, p,\nu\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p',\nu'\,\rangle,\ \langle\, q,\nu\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, q',\nu'\,\rangle}{\begin{array}{l}\langle\, p \parallel q,\nu\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p' \parallel q',\nu'\,\rangle\\ \langle\, p \mid q,\nu\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p' \parallel q',\nu'\,\rangle\end{array}}(16)$$

$$\frac{\langle\, p,\nu\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p',\nu'\,\rangle,\ \langle\, q,\nu\,\rangle\checkmark}{\begin{array}{l}\langle\, p \parallel q,\nu\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p',\nu'\,\rangle\\ \langle\, q \parallel p,\nu\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p',\nu'\,\rangle\\ \langle\, p \mid q,\nu\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p',\nu'\,\rangle\\ \langle\, q \mid p,\nu\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p',\nu'\,\rangle\end{array}}(17) \qquad \frac{\langle\, p,\nu\,\rangle \overset{a,\nu'}{\mapsto} \langle\, p',\nu''\,\rangle}{\begin{array}{l}\langle\, p \parallel q,\nu\,\rangle \overset{a,\nu'}{\mapsto} \langle\, p' \parallel q,\nu''\,\rangle\\ \langle\, q \parallel p,\nu\,\rangle \overset{a,\nu'}{\mapsto} \langle\, q \parallel p',\nu''\,\rangle\\ \langle\, p \between q,\nu\,\rangle \overset{a,\nu'}{\mapsto} \langle\, p' \parallel q,\nu''\,\rangle\end{array}}(18)$$

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

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

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

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

2.3 Robust Bisimilarity and Axiomatization

In this section the equivalence notion of robust bisimilarity is presented. First, the notion of bisimilarity of hybrid transition systems is defined. This notion is then lifted to process terms.

Definition 4 (Bisimilarity on hybrid transition systems)
Let $\langle X, A, \Sigma, \mapsto, \rightsquigarrow, \checkmark \rangle$ *be a hybrid transition system. A relation* $R \subseteq X \times X$ *on the state space, is a* bisimulation *relation if*

- *for all* $x, y \in X$ *such that* $x \,R\, y$*, we find* $\langle x \rangle \checkmark$ *implies* $\langle y \rangle \checkmark$*;*
- *for all* $x, y \in X$ *such that* $x \,R\, y$*, we find* $\langle y \rangle \checkmark$ *implies* $\langle x \rangle \checkmark$*;*
- *for all* $x, x', y \in X$ *such that* $x \,R\, y$ *and* $l \in A \cup \Sigma$*, we find* $\langle x \rangle \xrightarrow{l} \langle x' \rangle$ *implies there exists* y' *such that* $\langle y \rangle \xrightarrow{l} \langle y' \rangle$ *and* $x' \,R\, y'$*;*
- *for all* $x, y, y' \in X$ *such that* $x \,R\, y$ *and* $l \in A \cup \Sigma$*, we find* $\langle y \rangle \xrightarrow{l} \langle y' \rangle$ *implies there exists* x' *such that* $\langle x \rangle \xrightarrow{l} \langle x' \rangle$ *and* $x' \,R\, y'$*.*

Two states $x, y \in X$ *are* bisimilar*, notation* $x \leftrightarrows y$*, if there exists a bisimulation relation that relates* x *and* y*.*

In HyPA, model variables are shared by all processes executing in parallel. Therefore, a process can cause interference with another (parallel) process through these shared variables. In order for the equivalence to be robust with respect to this interference, it is required that process terms are related for all valuations that can be obtained through interference. An interference can be modeled as a function $\iota : Val \to Val$.

Definition 5 (Robust) *A relation* $R \subseteq (\mathcal{T}(\mathcal{V}_p) \times Val) \times (\mathcal{T}(\mathcal{V}_p) \times Val)$ *is called* robust *if for all* $\langle p, \nu \rangle, \langle p', \nu' \rangle \in X$ *such that* $\langle p, \nu \rangle \,R\, \langle p', \nu' \rangle$ *we find* $\langle p, \iota(\nu) \rangle \,R\, \langle p', \iota(\nu') \rangle$ *for all interferences* $\iota \in Val \to Val$*.*

Definition 6 (Robust bisimilarity) *Two process terms* $p, q \in \mathcal{T}(\mathcal{V}_p)$ *are robustly bisimilar, denoted* $p \leftrightarrows_r q$*, if there exists a robust bisimulation relation* $R \subseteq (\mathcal{T}(\mathcal{V}_p) \times Val) \times (\mathcal{T}(\mathcal{V}_p) \times Val)$ *such that* $\langle p, \nu \rangle \,R\, \langle q, \nu \rangle$ *for all valuations* $\nu \in Val$*.*

If two process terms are robustly bisimilar then they describe equivalent transition systems, hence they describe the same process. The axioms of HyPA are presented in table 4 below. The derivation rules of HyPA that define in which way equivalences can be derived from the axioms are given in [7, 6]. In each of the axioms, x, y, z denote arbitrary terms. The letters a, a' denote actions, while c, c' denote flow clauses and d, d' denote re-initialization clauses. Unlike what is usual for ACP, one may not choose δ when a is written in an axiom.

These axioms are sound with respect to robust bisimilarity and enable equational reasoning for the analysis of hybrid transition systems. Furthermore, using these axioms it is possible to rewrite a closed term into a normal form and it can be shown that HyPA is a conservative extension of the process algebra ACP [5, 7].

2.4 Example

In this section an example is given to illustrate HyPA's syntax. This example is a slightly modified version of the temperature controller in [13].

Table 4: Axioms of HyPA

$$
\begin{array}{lcl}
x \parallel y & \approx & x \between y \oplus y \between x \oplus x \mid y \\
x \blacktriangleright y & \approx & x \rhd y \oplus y
\end{array}
$$

$$
\begin{array}{lcl}
(x \oplus y) \oplus z & \approx & x \oplus (y \oplus z) \\
(x \rhd y) \rhd z & \approx & x \rhd (y \blacktriangleright z) \\
(x \mid y) \mid z & \approx & x \mid (y \mid z) \\
(x \oplus y) \odot z & \approx & x \odot z \oplus y \odot z \\
(x \oplus y) \between z & \approx & x \between z \oplus y \between z \\
x \oplus y & \approx & y \oplus x
\end{array}
\qquad
\begin{array}{lcl}
(x \odot y) \odot z & \approx & x \odot (y \odot z) \\
(x \between y) \between z & \approx & x \between (y \parallel z) \\
(x \mid y) \between z & \approx & x \mid (y \between z) \\
(x \oplus y) \rhd z & \approx & x \rhd z \oplus y \rhd z \\
(x \oplus y) \mid z & \approx & x \mid z \oplus y \mid z \\
x \mid y & \approx & y \mid x
\end{array}
$$

$$
\begin{array}{lcl}
x \odot \epsilon & \approx & x \\
\epsilon \rhd x & \approx & \epsilon \\
\epsilon \parallel x & \approx & x \\
(\mathit{false}) & \approx & \delta \\
[\mathit{false}] \gg x & \approx & \delta \\
c_{jmp} \gg c & \approx & c
\end{array}
\qquad
\begin{array}{lcl}
x \rhd \delta & \approx & x \\
\delta \mid x & \approx & \delta \\
\epsilon \between x & \approx & \delta \\
d \gg \delta & \approx & \delta \\
[\mathit{true}] \gg x & \approx & x
\end{array}
$$

$$
\begin{array}{lcl}
a \odot x \between y & \approx & a \odot (x \parallel y) \\
c \rhd x \between y & \approx & \delta
\end{array}
\qquad
\begin{array}{lcl}
a \odot x \rhd y & \approx & a \odot (x \blacktriangleright y) \\
(d \gg c \rhd x) \odot y & \approx & d \gg c \rhd x \odot y
\end{array}
$$

$$
\begin{array}{lcl}
d \gg x \oplus d' \gg x & \approx & (d \vee d') \gg x \\
d \gg (x \oplus y) & \approx & d \gg x \oplus d \gg y \\
(d \gg x) \rhd y & \approx & d \gg x \rhd y \\
d \gg (d' \gg x) & \approx & (d \sim d') \gg x
\end{array}
\qquad
\begin{array}{lcl}
(d \gg a) \odot x & \approx & d \gg a \odot x \\
(d \gg \epsilon) \odot x & \approx & d^? \gg x \\
d \gg x \between y & \approx & d \gg (x \between y)
\end{array}
$$

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

$$
\begin{array}{lcl}
d \gg \epsilon \mid d' \gg \epsilon & \approx & (d^? \wedge d'^?) \gg \epsilon \\
d \gg \epsilon \mid d' \gg a \odot x & \approx & \delta \\
d \gg a \odot x \mid d' \gg a' \odot y & \approx & (d \wedge d') \gg (a \gamma a') \odot (x \parallel y) \quad \text{if } a \gamma a' \text{ defined} \\
d \gg a \odot x \mid d' \gg a' \odot y & \approx & \delta \quad \text{if } a \gamma a' \text{ undefined} \\
d \gg \epsilon \mid d' \gg c \rhd x & \approx & (d^? \sim d') \gg c \rhd x \\
d \gg c \rhd x \mid d' \gg a \odot y & \approx & \delta \\
d \gg c \rhd x \mid d' \gg c' \rhd y & \approx & ((d \sim c_{jmp}) \wedge (d' \sim c'_{jmp})) \gg \\
 & & (c \wedge c') \rhd \left(\begin{array}{l} x \between c' \blacktriangleright y \ \oplus y \between c \blacktriangleright x \ \oplus \\ x \mid c' \blacktriangleright y \ \oplus y \mid c \blacktriangleright x \end{array} \right)
\end{array}
$$

The temperature in a room is controlled by a thermostat, which continuously monitors the temperature and turns a heater on and off. The thermostat has to keep the temperature x in the interval $[x_{min}, x_{max}]$. When the heater is turned off, the temperature decreases according to the differential equation $\dot{x} = -x$. When the heater is turned on, the temperature increases according to the differential equation $\dot{x} = -x + 4$. Initially, the heater is on.

$$
\begin{array}{lcl}
\text{HeaterOn} & \approx & (x|\, \dot{x} = -x + 4) \blacktriangleright \mathit{rOff} \odot \text{HeaterOff} \\
\text{HeaterOff} & \approx & (x|\, \dot{x} = -x) \blacktriangleright \mathit{rOn} \odot \text{HeaterOn} \\
\text{Thermostat} & \approx & (x_{min} \leq x \leq x_{max}) \blacktriangleright \left(\begin{array}{l} [x = x_{min}] \gg \mathit{sOn} \odot \text{Thermostat} \\ \oplus \\ [x = x_{max}] \gg \mathit{sOff} \odot \text{Thermostat} \end{array} \right) \\
\text{Controller} & \approx & \partial_H \left(\text{HeaterOn} \parallel \text{Thermostat}\right)
\end{array}
$$

$$
\begin{array}{lcl}
\gamma(\mathit{sOn}, \mathit{rOn}) & = & \mathit{On} \\
\gamma(\mathit{sOff}, \mathit{rOff}) & = & \mathit{Off} \\
H & = & \{\mathit{sOff}, \mathit{sOn}, \mathit{rOff}, \mathit{rOn}\}
\end{array}
$$

3 Abstraction of Model Variables

3.1 Syntax

HyPA is extended with an abstraction operator, which is denoted $[\![\ V\ \mid p\ ]\!]$. Here, V is a set of model variables, namely the variables to abstract from. Furthermore, p is an arbitrary HyPA process term, in which variables in V can occur as well as variables defined outside the scope of this abstraction operator (including the global model variables). Intuitively, the effect of abstracting from a certain variable is that this variable is not visible on the 'outside', i.e. the variable and its valuation cannot be observed externally. We often write $[\![\ v_1, \ldots, v_n\ \mid p\ ]\!]$ and $[\![\ v\ \mid p\ ]\!]$ instead of $[\![\ \{v_1, \ldots, v_n\}\ \mid p\ ]\!]$ and $[\![\ \{v\}\ \mid p\ ]\!]$, respectively.

Note that it is possible to use variable names in V that are already used outside the scope of this abstraction. For example, suppose there is a global model variable named v and $[\![\ v\ \mid p\ ]\!]$ is used. In this case, all occurrences of the name v in p bind to the abstracted variable v, not to the global variable v. The valuation of the abstracted variable v is not visible externally, but the valuation of the global variable v *is* visible.

3.2 Semantics

The semantics of the abstraction operator is defined using deduction rules in the style of [14]. Please refer to [7, 5] for a detailed description of the semantics of HyPA and the underlying hybrid transition system.

First, an auxiliary operator $|[\ V : \nu \mid p\ ]|$ is defined. The extra variable ν denotes the valuation of the variables in V, i.e. the local state, and its domain is exactly V. This auxiliary operator is needed to define the semantics of the normal abstraction operator $[\![\ V\ \mid p\ ]\!]$. It allows us to specify exactly what happens with the valuations of the abstracted variables and which valuations are visible in the underlying transition system. In principle it is possible to give axioms for this operator and use it in HyPA specifications, but so far this does not seem to be useful, especially for the purpose of linearization.

In table 5, the semantics of both types of abstraction operators is given. To keep the rules concise, the auxiliary functions $m_V(\mu, \nu)$ and $m_V(\sigma, \sigma')$ are used, which merge two valuations and two

Table 5: Operational semantics of the abstraction operator

$$\frac{\langle\, p, m_V(\mu,\nu)\,\rangle \checkmark}{\langle\, |[\ V : \nu \mid p\ ]|, \mu\,\rangle \checkmark}\ (1) \qquad \frac{\langle\, |[\ V : \nu \mid p\ ]|, \mu\,\rangle \checkmark}{\langle\, [\![\ V\ \mid p\ ]\!], \mu\,\rangle \checkmark}\ (4)$$

$$\frac{\langle\, p, m_V(\mu,\nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p', w'\,\rangle}{\langle\, |[\ V : \nu \mid p\ ]|, \mu\,\rangle \overset{a, m_V(w,\mu)}{\mapsto} \langle\, |[\ V : w'{\restriction_V}\ \mid p'\ ]|, m_V(w',\mu)\,\rangle}\ (2)$$

$$\frac{\langle\, p, m_V(\mu,\nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle}{\langle\, |[\ V : \nu \mid p\ ]|, \mu\,\rangle \overset{m_V(\sigma,\sigma')}{\rightsquigarrow} \langle\, |[\ V : w'{\restriction_V}\ \mid p'\ ]|, m_V(w',\sigma'(\uparrow))\,\rangle}\ (3)$$

$$\frac{\langle\, |[\ V : \nu \mid p\ ]|, \mu\,\rangle \overset{a,w}{\mapsto} \langle\, p', w'\,\rangle}{\langle\, [\![\ V\ \mid p\ ]\!], \mu\,\rangle \overset{a,w}{\mapsto} \langle\, p', w'\,\rangle}\ (5) \qquad \frac{\langle\, |[\ V : \nu \mid p\ ]|, \mu\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle}{\langle\, [\![\ V\ \mid p\ ]\!], \mu\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle}\ (6)$$

flows, respectively. The first function takes the valuations of the variables in the set V from ν and the valuations of all other variables from μ. Similarly, the second one takes the flows of the variables in the set V from σ' and the flows of all other variables from σ. Intuitively, this works like opening a new variable scope in a (procedural) programming language, because the newly introduced variables 'hide' existing variables with the same name. For any $\mu, \nu \in \mathit{Val}$, $\sigma, \sigma' \in \Sigma$ and $V \subseteq \mathcal{V}_{\mathrm{m}}$:

$$m_V(\mu,\nu)(n) = \begin{cases} \mu(n) & \text{if } n \notin V \\ \nu(n) & \text{if } n \in V \end{cases} \qquad m_V(\sigma,\sigma')(n) = \begin{cases} \sigma(n) & \text{if } n \notin V \\ \sigma'(n) & \text{if } n \in V \end{cases}$$

Furthermore, $\nu{\restriction_V}$ denotes the valuation where $\mathit{dom}(\nu{\restriction_V}) = V$ and $\nu{\restriction_V}\,(n) = \nu(n)$ for all $n \in V$. Finally, $\sigma'(\uparrow)$ denotes the valuation of the flows in σ' in the last element of its domain.

Rule (1) states that the abstraction of a terminating process can also terminate. Rule (2) describes the case for a (discrete) action transition. This rule expresses an essential point of the abstraction operator, namely that the valuations of abstracted variables are not visible in the transition system. That is why the arrow in the conclusion is labeled with $m_V(w,\mu)$, instead of simply w. Furthermore, the semantics is chosen such that the valuation of 'hidden' model variables does not change during an action transition. The reason for this choice is that in the existing semantics of HyPA, the valuations of model variables also do not change during an action transition. Rule (3) describes the case for a (continuous) flow transition. This rule is similar to rule (2). Note however that the valuation in the resulting state is equal to the last valuation of the flow. Again, the reason for this choice is that in the existing semantics of HyPA, this is also the case. Rules (4) to (6) define the actual abstraction operator, in terms of the auxiliary abstraction operator. Note that the local state variable ν in the hypotheses of these rules is an arbitrary valuation whose domain is V.

Theorem 1 *Robust bisimilarity is a congruence for both the auxiliary abstraction operator* $|[\ V : \nu \mid p\ ]|$ *and the abstraction operator* $[\![\ V\ \mid p\ ]\!]$. *Hence, if* $p \leftrightarrows_r q$, *then* $|[\ V : \nu \mid p\ ]| \leftrightarrows_r |[\ V : \nu \mid q\ ]|$ *and* $[\![\ V\ \mid p\ ]\!] \leftrightarrows_r [\![\ V\ \mid q\ ]\!]$ *for all process terms* p *and* q, $V \subseteq \mathcal{V}_{\mathrm{m}}$ *and* $\nu \in \mathit{Val}$.

Proof This is straightforward to verify using the congruence formats in [12]. ⊠

3.3 Axioms

A (partial) axiomatization is given in table 6. In these axioms, $Var(x)$ denotes the set of free variables in the term x. Table 6 only gives some basic axioms and some axioms that are necessary for the linearization algorithm.

Table 6: Axioms for the abstraction operator

$[\![V \mid x]\!] \oplus [\![V \mid y]\!]$	$\approx$	$[\![V \mid x \oplus y]\!]$	(VA1)
$[\![V \mid x]\!] \odot [\![V \mid y]\!]$	$\approx$	$[\![V \mid x \odot [V \mid true] \gg y]\!]$	(VA2)
$[\![V \mid x]\!] \rhd [\![V \mid y]\!]$	$\approx$	$[\![V \mid x \rhd [V \mid true] \gg y]\!]$	(VA3)
$[\![V \mid \partial_H(x)]\!]$	$\approx$	$\partial_H([\![V \mid x]\!])$	(VA4)
$[\![V \mid [\![W \mid x]\!]]\!]$	$\approx$	$[\![V \cup W \mid x]\!]$	(VA5)
$[\![V \mid x]\!]$	$\approx$	x, if $Var(x) \cap V = \emptyset$	(VA6)
$[\![V \mid x]\!] \parallel y$	$\approx$	$[\![V \mid x \parallel y]\!]$, if $Var(y) \cap V = \emptyset$	(VA7)
$d \gg [\![V \mid x]\!]$	$\approx$	$[\![V \mid d \gg x]\!]$, if $Var(d) \cap V = \emptyset$	(VA8)
$[\![v \mid x]\!]$	$\approx$	$[\![w \mid x[^w/_v]]\!]$, if $w \notin Var(x)$	(VA9)

Axioms (VA1), (VA2), (VA3) and (VA4) all express distribution of the abstraction operator over other operators. Together, they describe how abstraction can be distributed over (closed) linear terms. Axiom (VA5) describes how two abstractions can be merged. Axiom (VA6) states that abstracting from variables that do not occur freely in the abstracted term has no influence. Both of these axioms are useful for introducing abstractions or for eliminating them. Axioms (VA7) and (VA8) expresses that a parallel term or re-initialization clause can be pulled into the abstraction, as long as the abstracted variables do not occur freely in that term or clause. These axioms appeal to the same intuition as axiom (VA6). However, they cannot simply be derived from (VA6), because abstraction does not distribute over parallel composition or re-initialization in general. Axiom (VA9) states that abstracted variables can be renamed (α-conversion). Note that in this axiom v and w denote single variables, not sets of variables. The expression $x[^w/_v]$ denotes the substitution of w for all free occurrences of v in the process term x.

Theorem 2 *All axioms in table 6 are sound.*

Proof See appendix A for the soundness proofs. ⊠

4 Linearization

This section describes an algorithm for linearization of (a large subset of) HyPA. First, the specification of a general HyPA linearization algorithm is given. Second, the input restrictions of our particular algorithm are defined and the specification of a general linearization algorithm is adapted to these input restrictions. Third, the linearization algorithm is described informally and the formal translation functions are presented and the fact that they preserve equivalence is proven.

4.1 Specification of a General HyPA Linearization Algorithm

In this section, first the notions *recursive specification* and *linear recursive specification* are defined. Then, the specification of a general HyPA linearization algorithm is stated.

Definition 7 (Recursive Specification) *A recursive specification consists of an open term t, called the* initial term*, and a finite set of* recursive equations *E that define the variables in t. Such a specification is denoted $\langle\, t \mid E \,\rangle$. A recursive equation for a (recursion) variable X is defined as $X \approx s(Var(E))$, meaning that the right-hand side of such an equation is an open term s that only contains variables that are defined in the specification. There is exactly one recursive equation for every recursion variable.*

Definition 8 (Solutions of a Recursive Specification) *A solution of a recursive specification $\langle\, t \mid E \,\rangle$ is a collection of processes that can be substituted for the recursion variables, such that all recursive equations in E become true statements. The variables of the initial term t are then interpreted as their solutions in E. For example, a solution of the recursive equation $X \approx b \odot X$ is the process $b \odot b \odot b \odot \ldots$. A solution of the specification $\langle\, a \odot X \mid \{X \approx b \odot X\} \,\rangle$ is then $a \odot (b \odot b \odot b \odot \ldots)$.*

Definition 9 (Linear Recursive Specification) *A linear recursive specification $\langle\, t \mid E \,\rangle$ is a recursive specification that satisfies the following requirements:*

1. *the initial term t is either a recursion variable in E (i.e. X) or the abstraction of a re-initialization on such a variable (i.e. $[\![\, V \mid d \gg X \,]\!]$), and*
2. *the right-hand sides of all recursive equations in E are* linear terms*.*

A term is linear, if it has the form[1]

$$p \quad ::= \quad \delta \;\mid\; d \gg a \;\mid\; d \gg a \odot X \;\mid\; d \gg c \rhd X \;\mid\; p \oplus p$$

and the right-hand sides of all recursion variables occurring in the term are linear as well.

Now, a general HyPA linearization algorithm is specified as follows:

Specification 1 *A HyPA linearization algorithm transforms a recursive specification $\langle\, t \mid E \,\rangle$ into a linear recursive specification $\langle\, t' \mid E' \,\rangle$ such that $\langle\, t \mid E \,\rangle$ and $\langle\, t' \mid E' \,\rangle$ are robustly bisimilar.*

4.2 Input Restrictions

The linearization algorithm presented in this paper has some restrictions on the input specifications. First some definitions are stated, which are then used to express these restrictions.

Definition 10 (Guarded process term) *An open process term p is* guarded *if all occurrences of recursion variables in p are in the scope of an action prefix $a \odot \_$ or a flow prefix $c \rhd \_$.*

[1] Forms of process terms are denoted in the familiar BNF notation. Recall that X denotes a recursion variable, a is a discrete action, c is a flow clause and d is a re-initialization clause.

Definition 11 (Guarded recursive specification) *A recursive specification $\langle t \mid E \rangle$ is guarded if the right-hand sides of all equations in E are guarded, or can be transformed into guarded terms by replacing variables by the right-hand side of their equation*[2]*.*

Note that the Recursive Specification Principle (RSP) and Recursive Definition Principle (RDP) from [7, 6] together state that guarded recursive specifications have a unique solution.

Definition 12 ($\mathbf{HyPA_{par}}$ Form) *The $HyPA_{\mathrm{par}}$ form is defined as the form p:*

$$\begin{array}{lcl} p & ::= & [\![V \mid q]\!] \mid q \\ q & ::= & X \mid q \parallel q \mid \partial_H(q) \end{array}$$

Definition 13 ($\mathbf{HyPA_{lin}}$ Form) *The $HyPA_{\mathrm{lin}}$ form is defined as follows:*

$$\begin{array}{lcl} p & ::= & a \mid X \mid \delta \mid c \mid p \oplus p \mid p \odot p \mid \\ & & d \gg p \mid c \blacktriangleright p \mid c \rhd p \end{array}$$

Definition 14 ($\mathbf{HyPA_{lin}}$ Specification) *A* $\mathrm{HyPA_{lin}}$ specification *is a recursive specification $\langle t \mid E \rangle$ that satisfies the following three restrictions:*

1. *$\langle t \mid E \rangle$ is guarded;*
2. *t is in $HyPA_{\mathrm{par}}$ form, and*
3. *the right-hand sides of all recursive equations in E are in $HyPA_{\mathrm{lin}}$ form.*

Specification 2 *The linearization algorithm that is presented in this paper transforms a $HyPA_{\mathrm{lin}}$ specification $\langle t \mid E \rangle$ into a linear recursive specification $\langle t' \mid E' \rangle$, such that $\langle t \mid E \rangle$ and $\langle t' \mid E' \rangle$ are robustly bisimilar.*

Most restrictions are made in order to avoid fundamental problems. First, the parallel composition is restricted in such a way that there is no recursion over the parallel composition, as in $X \approx X \parallel Y$ for instance. In such a case, there are in fact infinitely many parallel compositions, so trying to eliminate them one by one does not work[3]. Second, the abstraction operator is not allowed in the $\mathrm{HyPA_{lin}}$ form, because it is not possible to eliminate abstraction of open terms from recursive equations. Third, the empty process (ϵ) cannot be used, because it leads to some problems in the transformations.

Finally, only single flow clauses can be disrupted. On the one hand, this is a consequence of not allowing the use of the empty process, because the empty process is needed when there are actions in the left argument of a disrupt operator. For instance, eliminating the disrupt operator from $a \blacktriangleright x$ yields $a \odot (\epsilon \oplus x) \oplus x$ and the empty process can not be eliminated from this term in general. On the other hand, recursion in the left argument of the disrupt operator gives problems similar to the problems with recursion in the scope of parallel composition. Every time a disrupt operator is eliminated, another may be introduced. For instance, $X \approx (a \odot X) \blacktriangleright y \approx (a \odot X) \rhd y \oplus y \approx a \odot (X \blacktriangleright y) \oplus y$. Clearly, trying to eliminate the disrupt operators one by one does not work in this case.

[2]The usual definition of guardedness states that a recursive specification is also guarded if it can be transformed into a guarded recursive specification by applying axioms. In this paper however, specifications that need to be transformed are not considered to be guarded, otherwise correctness of our algorithm cannot be proven.

[3]In the linearization algorithm of μCRL a similar restriction on the use of both the parallel composition and the encapsulation operator was made at first. The algorithm for μCRL was later extended to relax this restriction, but it became much more complex. We decided to take the same approach and keep the restriction for now.

However, a few of the restrictions are only made to keep the description of the algorithm concise. First, as noted above, only a single flow clause is allowed in the left argument of a disrupt operator. It is straightforward to allow alternative and sequential composition of flow clauses, re-initializations, disrupt operators and deadlock (δ) here, since the problem is the use of actions and recursion variables. Second, only one abstraction operator is allowed in the $\mathrm{HyPA}_{\mathrm{par}}$ form. Third, the encapsulation operator is not allowed in the $\mathrm{HyPA}_{\mathrm{lin}}$ form. Finally, the left-parallel composition and forced-synchronization are not allowed at all, mainly because they are not meant to be used directly in specifications. The first three restrictions are easily relaxed and could be implemented as some sort of pre-processing step to the algorithm described in this paper. The last one can partly be relaxed easily as well, but it gives the same problems with recursion as the parallel composition.

4.3 Linearization Algorithm

The algorithm consists of three consecutive stages. The first two stages deal with the recursive equations in the input specification and the third stage deals with the initial term. In sections 4.3.1 to 4.3.3, each of these three stages is described in detail. In this section, correctness of our algorithm is proven as well, which is captured in the following theorem:

Theorem 3 *The linearization algorithm satisfies Specification 2.*

Proof This theorem leads to three proof obligations. First, the algorithm has to be *sound*, which means that all transformation steps transform the specification into another specification whose initial terms are robustly bisimilar. Second, the *result* of the algorithm has to be a linear recursive specification. Finally, the algorithm has to be *well-defined*.

In the following, it is proved for each of the three stages that they are sound and that they are well-defined. Therefore, the algorithm as a whole is sound and well-defined.

In the following sections it is also proved that the result of the algorithm is a linear recursive specification. The input of the very first step is a $\mathrm{HyPA}_{\mathrm{lin}}$ specification. Each step is proven to lead to a specification of a certain intermediate form, and this intermediate form is then the input for the next step. Finally, the very last step (T_1) is shown to lead to a recursive specification of linear form.

Therefore, the linearization algorithm presented in this paper satisfies Specification 2. ⊠

4.3.1 Stage 1: Transforming Equations into Semi-linear Form

In this first stage, several transformation functions are applied to the right-hand sides of all recursive equations in the specification. These transformation functions are applied in the order in which they are described below.

The form of the right-hand sides after this first stage is called *semi-linear*, which is defined as follows:

$$\begin{array}{lcl} p & ::= & \delta \mid d \gg a \mid d \gg a \odot q \mid d \gg c \rhd q \mid p \oplus p \\ q & ::= & X \mid q \odot q \end{array}$$

The only difference with the linear form is the fact that an action a or a flow clause c can be followed by the sequential composition of multiple recursion variables (q), instead of a single variable (X) only.

Simple Rewriting This first step eliminates the disrupt operator ($\blacktriangleright$) and distributes the alternative composition ($\oplus$) over sequential composition and re-initialization. This is done by rewriting the right-hand sides of all recursive equations in E using the rewrite system consisting of the following rewrite rules:

$$\begin{array}{lcll} x \blacktriangleright y & \to & x \rhd y \oplus y & \text{(R1)} \\ (x \oplus y) \odot z & \to & x \odot z \oplus y \odot z & \text{(R2)} \\ d \gg (x \oplus y) & \to & d \gg x \oplus d \gg y & \text{(R3)} \end{array}$$

Recall that the right-hand sides of all recursive equations in E are of the form $\mathrm{HyPA}_{\mathrm{lin}}$:

$$\begin{array}{lll} p & ::= & a \mid X \mid \delta \mid c \mid p \oplus p \mid p \odot p \mid \\ & & d \gg p \mid c \blacktriangleright p \mid c \rhd p \end{array}$$

Lemma 1 (Resulting form) *After applying this rewrite system to the right-hand sides of all equations in E, the right-hand sides of all equations in E are of the form r:*

$$\begin{array}{lll} r & ::= & a \mid X \mid \delta \mid c \mid d \gg r_2 \mid r_2 \odot r \mid c \rhd r \mid r \oplus r \\ r_2 & ::= & a \mid X \mid \delta \mid c \mid d \gg r_2 \mid r_2 \odot r \mid c \rhd r \end{array}$$

Proof We prove that normal forms of this rewrite system, when applied to terms of the form $\mathrm{HyPA}_{\mathrm{lin}}$, are of the form r. The atoms of the $\mathrm{HyPA}_{\mathrm{lin}}$ form (a, X, δ, c) are in r. Therefore, every $\mathrm{HyPA}_{\mathrm{lin}}$ term $p \notin r$ has a smallest sub term $s \notin r$ of the form R, where r denotes the form r:

$$R \quad ::= \quad r \oplus r \mid r \odot r \mid d \gg r \mid c \blacktriangleright r \mid c \rhd r$$

We give at least one applicable rewrite rule for every of these possible sub terms, unless the specific sub term is in normal form already. In that case, we do not need to give a rule, because we have a contradiction with the initial assumption that $s \notin r$.

- s of the form $r \oplus r$: implies $s \in r$.
- s of the form $r \odot r$:
  - s of the form $a \odot r$, $X \odot r$, $\delta \odot r$, $c \odot r$, $(d \gg r_2) \odot r$, $(r_2 \odot r) \odot r$ or $(c \rhd r) \odot r$: implies $s \in r$.
  - s of the form $(r \oplus r) \odot r$: rewrites using (R2).
- s of the form $d \gg r$:
  - s of the form $d \gg a$, $d \gg X$, $d \gg \delta$, $d \gg c$, $d \gg (d \gg r_2)$, $d \gg (r_2 \odot r)$ or $d \gg (c \rhd r)$: implies $s \in r$.
  - s of the form $d \gg (r \oplus r)$: rewrites using (R3).
- s of the form $c \blacktriangleright r$: rewrites using (R1).
- s of the form $c \rhd r$: implies $s \in r$.

All existing equations in E are rewritten into the form r and no new equations are added to E. Therefore, after applying this rewrite system to the right-hand sides of all equations in E, the right-hand sides of all equations in E are of the form r. ⊠

Lemma 2 (Soundness) *This rewrite step is sound.*

Proof All rewrite rules are directed version of HyPA axioms. Therefore, this rewrite step is sound. ⊠

Lemma 3 (Termination) *This rewrite system is terminating.*

Proof In [5], a rewrite system is presented that is proven to rewrite closed HyPA terms into basic terms. That rewrite system was proved to be terminating. The rewrite system used in this step is a subset of that rewrite system, so the rewrite system in this step is terminating as well. ⊠

Adding New Recursive Equations This second step reduces the complexity of the terms further, by introducing new recursive equations. The right-hand sides become terms that are the alternative composition of sub terms (summands) that do not contain alternative composition themselves. Furthermore, they either have a single left disrupt operator or a number of sequential compositions, but not both.

The right-hand sides of all recursive equations in E are transformed with the function S_1. The function S_1 is not applied to the right-hand sides of newly introduced equations, except when this is stated explicitly. The introduction of a new recursive equation is denoted by adding it to the set of recursive equations E. Note that it is always assumed that the left-hand side of this new equation is a fresh variable.

$$
\begin{array}{lcl}
S_1(a) & = & a \\
S_1(X) & = & X \\
S_1(\delta) & = & \delta \\
S_1(c) & = & X, \text{ and } E := E \cup \{X \approx c \rhd Y, Y \approx \delta\} \\
S_1(d \gg p) & = & d \gg S_1(p) \\
S_1(p \odot q) & = & S_2(p \odot q) \\
S_1(c \rhd p) & = & S_2(c \rhd p) \\
S_1(p \oplus q) & = & S_1(p) \oplus S_1(q)
\end{array}
$$

$$
\begin{array}{lcl}
S_2(a) & = & a \\
S_2(X) & = & X \\
S_2(\delta) & = & \delta \\
S_2(c) & = & X, \text{ and } E := E \cup \{X \approx c \rhd Y, Y \approx \delta\} \\
S_2(d \gg p) & = & d \gg S_2(p) \\
S_2(p \odot q) & = & X, \text{ and } E := E \cup \{X \approx S_2(p) \odot S_2(q)\} \\
S_2(c \rhd p) & = & X, \text{ and } E := E \cup \{X \approx c \rhd Y, Y \approx S_1(p)\} \\
S_2(p \oplus q) & = & X, \text{ and } E := E \cup \{X \approx S_1(p) \oplus S_1(q))\}
\end{array}
$$

Lemma 4 (Resulting form) *After applying S_1 to the right-hand sides of all equations in E, the right-hand sides of all equations in E are of the form s:*

$$
\begin{array}{lcl}
s & ::= & a \mid X \mid \delta \mid d \gg s_2 \mid s_2 \odot s_2 \mid c \rhd X \mid s \oplus s \\
s_2 & ::= & a \mid X \mid \delta \mid d \gg s_2
\end{array}
$$

Proof Before applying S_1, the right-hand sides of all equations in E have the form r. We have two proof obligations.

First, $S_1(p)$ is of the form s, for any term p of the form r: applying the function S_1 to the form r gives the following form. The sub form s_1^1 results from S_1 applied to r, s_2^1 results from S_1 applied to r_2, s_1^2 results from S_2 applied to r, and s_2^2 results from S_2 applied to r_2.

$$
\begin{array}{lcl}
s_1^1 & ::= & a \mid X \mid \delta \mid c \rhd X \mid d \gg s_2^1 \mid s_2^2 \odot s_1^2 \mid s_1^1 \oplus s_1^1 \\
s_2^1 & ::= & a \mid X \mid \delta \mid d \gg s_2^1 \\
s_1^2 & ::= & a \mid X \mid \delta \mid d \gg s_2^2 \\
s_2^2 & ::= & a \mid X \mid \delta \mid d \gg s_2^2
\end{array}
$$

We see that s_2^1, s_1^2 and s_2^2 are identical[4]. Therefore, we can delete two of these rules and replace all references to the two deleted rules by the name of the third remaining rule. Renaming the two rules that are left gives the form s.

Second, all newly introduced equations are of the form s: equations with right-hand sides of the following form might be introduced:

- δ, which is in s.
- $c \rhd Y$, which is in s.
- $S_1(p)$ for p of the form r: $S_1(p)$ is of the form s.
- $S_1(p) \oplus S_1(q)$ for p and q of the form r: $S_1(p)$ and $S_1(q)$ are both of the form s and $s \oplus s$ is in the form s.
- $S_2(p) \odot S_2(q)$ for p of the form r_2 and q of the form r: First, we prove by induction that $S_2(x)$ is of the form s_2 for any x of the form r:
  - a: $S_2(a) = a$, which is in s_2.
  - δ: $S_2(\delta) = \delta$, which is in s_2.
  - X, c, $y \odot z$, $c \rhd z$, $z \oplus z$, where y is of the form r_2 and z is of the form r: applying S_2 to these terms gives a single recursion variable X, which is in s_2.
  - $d \gg y$, where y is of the form r_2: $S_2(d \gg y) = d \gg S_2(y)$. Since r_2 is a subset of r, applying the induction hypothesis gives $d \gg S_2(y) = d \gg z$ where z is of the form s_2. $d \gg y$ is of the form s_2 as well, so $S_2(d \gg y)$ is of the form s_2.

  Now, we see that $S_2(p)$ and $S_2(q)$ are both of the form s_2. $s_2 \odot s_2$ is in s, so $S_2(p) \odot S_2(q)$ is of the form s.

We conclude that after applying S_1 to the right-hand sides of all equations in E, the right-hand sides of all equations in E are of the form s. ⊠

Lemma 5 (Soundness) *S_1 is sound, i.e. $S_1(p) \approx p$, for any term p of the form r.*

Proof First, we prove by induction on the form r that $S_1(p) \approx S_2(p)$ for any p of the form r.
Base cases:

- a, X, δ: trivial.
- c: $S_1(c) = X \approx c \rhd Y \approx c \rhd \delta \approx c$ and $S_2(c) = X \approx c \rhd Y \approx c \rhd \delta \approx c$, so $S_1(c) \approx S_2(c)$.
- $p \odot q$, where p is of the form r_2 and q is of the form r: from the definition of S_1 follows that $S_1(p \odot q) = S_2(p \odot q)$.
- $c \rhd p$, where p is of the form r: from the definition of S_1 follows that $S_1(c \rhd p) = S_2(c \rhd p)$.
- $p \oplus q$, where p and q are of the form r: $S_2(p \oplus q) = X \approx S_1(p) \oplus S_1(q) = S_1(p \oplus q)$.

Induction case:

- $d \gg p$, where p is of the form r_2: r_2 is a subset of r, so using the induction hypothesis $S_1(p) \approx S_2(p)$ we see that $S_1(d \gg p) = d \gg S_1(p) \approx d \gg S_2(p) = S_2(d \gg p)$.

Now, we prove by induction that $S_2(p) \approx p$ for any p of the form r.
Base cases:

[4] Although $d \gg s_2^1$ and $d \gg s_2^2$ appear to be different, they describe the same form in these three BNF rules.

- a, X, δ: trivial.
- c: $S_2(c) = X \approx c \rhd Y \approx c \rhd \delta \approx c$.

Induction cases:

- $d \gg p$, where p is of the form r_2: using the induction hypothesis $S_2(p) \approx p$, we see that $S_2(d \gg p) = d \gg S_2(p) \approx d \gg p$.
- $p \odot q$, where p is of the form r_2 and q is of the form r: using the induction hypothesis twice gives us $S_2(p \odot q) = X \approx S_2(p) \odot S_2(q) \approx p \odot q$.
- $c \rhd p$, where p is of the form r: using the fact that $S_1(p) \approx S_2(p)$ and the induction hypothesis we see that $S_2(c \rhd p) = X \approx c \rhd Y \approx c \rhd S_1(p) \approx c \rhd S_2(p) \approx c \rhd p$.
- $p \oplus q$, where p and q are of the form r: using the fact that $S_1(p) \approx S_2(p)$ and the induction hypothesis we see that $S_2(p \oplus q) = X \approx S_1(p) \oplus S_1(q) \approx S_2(p) \oplus S_2(q) \approx p \oplus q$.

From $S_1(p) \approx S_2(p)$ and $S_2(p) \approx p$ we conclude that $S_1(p) \approx p$. Therefore, S_1 is sound. ⊠

Lemma 6 (Well-definedness) *S_1 is well-defined.*

Proof When either of the functions S_1 or S_2 is used in the right-hand side, it is usually applied to a (strict) sub term of the argument of the left-hand side. One exception is the case $p \odot q$ in S_1, but here we see that $S_1(p \odot q) = S_2(p \odot q) = X$ and $E := E \cup \{X \approx S_2(p) \odot S_2(q)\}$, so immediately after the next step there is recursion on strictly smaller sub terms of $p \odot q$. The only other exception is $c \rhd p$ in S_1, which is analogous to the previous exception. Therefore, S_1 is well-defined. ⊠

Guarding This third step transforms all right-hand sides of all recursive equations in E into guarded terms. Recall that the definition of guardedness states that a term is guarded if all recursion variables occur in the scope of an action prefix $a \odot \_$ or a flow prefix $c \rhd \_$. However, a recursive specification is guarded if the right-hand sides of all equations are guarded, *or* can be transformed into guarded terms by replacing variables by the right-hand side of their equation. This means that, although the specification is guarded, the right-hand sides of the recursive equations may not be guarded.

The right-hand sides of all recursive equations in E are transformed with the function *guard*:

$$\begin{array}{lcl}
guard(a) & = & a \\
guard(X) & = & guard(rhs(X)) \\
guard(\delta) & = & \delta \\
guard(d \gg p) & = & rewr(d \gg guard(p)) \\
guard(p \odot q) & = & rewr(guard(p) \odot q) \\
guard(c \rhd p) & = & c \rhd p \\
guard(p \oplus q) & = & guard(p) \oplus guard(q)
\end{array}$$

where $rhs(X)$ denotes the right-hand side of the recursive equation of X in E, and *rewr* is the

rewrite system consisting of the following rules:

$$
\begin{array}{lcll}
(x \oplus y) \odot z & \to & x \odot z \oplus y \odot z & \text{(R10)} \\
d \gg (x \oplus y) & \to & d \gg x \oplus d \gg y & \text{(R11)} \\
(c \rhd x) \odot y & \to & c \rhd (x \odot y) & \text{(R12)} \\
d \gg \delta & \to & \delta & \text{(R14)} \\
\delta \odot x & \to & \delta & \text{(R15)} \\
(d \gg (c \rhd x)) \odot y & \to & d \gg c \rhd (x \odot y) & \text{(R16)} \\
d \gg (d' \gg x) & \to & (d \sim d') \gg x & \text{(R17)} \\
(x \odot y) \odot z & \to & x \odot (y \odot z) & \text{(R18)} \\
(d \gg (a \odot x)) \odot y & \to & d \gg (a \odot (x \odot y)) & \text{(R19)} \\
d \gg ((d' \gg a) \odot x) & \to & (d \sim d') \gg (a \odot x) & \text{(R20)}
\end{array}
$$

Definition 15 (PNUDG) *The Process Name Unguarded Dependency Graph (PNUDG)*[5] *of a specification* $\langle t \mid E \rangle$ *is constructed as follows. Every recursion variable in the specification is a node and there is a directed edge from a node* X *to a node* Y *if* Y *occurs unguarded in the right-hand side of the equation for* X *in* E*.*

Lemma 7 *If the specification* $\langle t \mid E \rangle$ *is guarded, then its PNUDG has no cycles.*

Proof Recall that the input of the linearization algorithm is a *guarded* recursive specification. A recursive specification $\langle t \mid E \rangle$ is guarded if the right-hand sides of all equations in E are guarded, *or* can be transformed into guarded terms by replacing variables by the right-hand side of their equation.

Suppose $\langle t \mid E \rangle$ is guarded. Then, we have the following two cases:

- the right-hand sides of all equations in E are guarded: the PNUDG contains no edges, so the PNUDG has no cycles.
- the right-hand sides of all equations in E can be transformed into guarded terms by replacing variables by the right-hand side of their equation: to make a certain equation X in E guarded, we have to repeatedly substitute unguarded occurrences of recursion variables by the right-hand side of their defining equations in the right-hand side of the equation for X. This means that, to make X guarded, we at least have to follow every path in the PNUDG starting from X. The fact that it is possible to make X guarded, implies that it takes only finitely many substitutions to do that. Therefore all paths in the PNUDG starting in X are finite. This holds for every equation in E, so all paths in the PNUDG are finite. Therefore, the PNUDG has no cycles.

⊠

Lemma 8 (Well-definedness) *guard is well-defined.*

Proof In [5], a rewrite system is presented that is proven to rewrite closed HyPA terms into basic terms. That rewrite system was also proved to be terminating. The rewrite system *rewr* is a subset of that rewrite system, so *rewr* is terminating as well.

The only case that makes the argument of *guard* larger is the case of X. Note that $guard(X)$ is only applied to unguarded occurrences of X, because guarded occurrences of recursion variables

[5] This definition is adapted from [16], but our notion of guardedness is quite different.

only occur in the right argument of the sequential composition ($\odot$) and the *guard* function is not applied to these arguments ($guard(p \odot q) = rewr(guard(p) \odot q)$). Now, let n be the number of equations in E. Due to the fact that the PNUDG is acyclic, this clause cannot be applied more than n times (otherwise there would have to be a cycle in the PNUDG). ⊠

Lemma 9 *$rewr(p)$ is of the form g, for any p of the form g.*

Proof We prove this by showing that each rewrite rule has this property. If a rule is not applicable to any term of the form g, then it follows trivially that this term is still of the form g.

- (R10), (R11), (R12), (R14), (R16), (R17), (R19) and (R20): cannot be applied to any term of the form g.
- (R15): this rule is only applicable to sub terms of the form g_2. Application of this rule gives δ, which is in g_2 as well. We replaced a g_2 sub term by another g_2 term, so the resulting term is still in g.
- (R18): this rule is only applicable to sub terms of the form g_2, where x, y and z are of the form g_2. Application of this rule gives $x \odot (y \odot z)$, which is in g_2 as well. We replaced a g_2 sub term by another g_2 term, so the resulting term is still in g.

⊠

Lemma 10 *$rewr(d \gg p)$ has the form g, for any p of the form g.*

Proof We prove this by induction on the structure of the form g. We show for any term of the form $d \gg p$, where p is of the form g, that it is already in g or can be rewritten into a term that is in g. Note that according to lemma 9, such a term will remain in g after applying other rewrite rules. Also note that rewriting on p itself has no influence, because p will remain in g.

- $d \gg (q \oplus r)$, for q, r in g: only rewrites with (R11) to $d \gg q \oplus d \gg r$. Applying the induction hypothesis twice gives that $rewr(d \gg q)$ is in g and $rewr(d \gg r)$ is in g. $g \oplus g$ is in g, so $rewr(d \gg (q \oplus r))$ is in g.
- $d \gg \delta$: only rewrites with (R14) to δ, which is in g.
- $d \gg a$: is in g.
- $d \gg (a \odot q)$, for q in g_2: is in g.
- $d \gg (c \rhd q)$, for q in g_2: is in g.
- $d \gg (d' \gg a)$: only rewrites with (R17) to $(d \sim d') \gg a$, which is in g.
- $d \gg (d' \gg (a \odot q))$, for q in g_2: only rewrites with (R17) to $(d \sim d') \gg (a \odot q)$, which is in g.
- $d \gg ((d' \gg a) \odot q)$, for q in g_2: only rewrites with (R20) to $(d \sim d') \gg (a \odot q)$, which is in g.
- $d \gg (d' \gg (c \rhd q))$, for q in g_2: only rewrites with (R17) to $(d \sim d') \gg (c \rhd q)$, which is in g.

⊠

Lemma 11 *$rewr(p \odot q)$ has the form g, for p of the form g and q of the form s_2.*

Proof We prove this by induction on the structure of the form g. We show for any term of the form $p \odot q$, where p is of the form g and q is of the form s_2, that it is already in g or can be rewritten into a term that is in g. Note that according to lemma 9, such a term will remain in g after applying other rewrite rules. Also note that rewriting on p itself has no influence, because p will remain in g and it can be verified that the same holds for q.

- $(r \oplus s) \odot q$, for r, s in g: rewrites with (R10) to $r \odot q \oplus s \odot q$. Applying the induction hypothesis twice gives that $rewr(r \odot q)$ is in g and $rewr(s \odot q)$ is in g. $g \oplus g$ is in g, so $r \odot q \oplus s \odot q$ is in g.
- $\delta \odot q$: rewrites with (R15) to δ, which is in g.
- $a \odot q$: s_2 is a subset of g_2 and $a \odot g_2$ is in g, so $a \odot q$ is in g.
- $(a \odot r) \odot q$, for r in g_2: rewrites with (R18) to $a \odot (r \odot q)$. s_2 is a subset of g_2 and $a \odot (g_2 \odot g_2)$ is in g, so $a \odot (r \odot q)$ is in g.
- $(c \rhd r) \odot q$, for r in g_2: rewrites with (R12) to $c \rhd (r \odot q)$. s_2 is a subset of g_2 and $c \rhd (g_2 \odot g_2)$ is in g, so $c \rhd (r \odot q)$ is in g.
- $(d \gg a) \odot q$: s_2 is a subset of g_2 and $(d \gg a) \odot g_2$ is in g, so $(d \gg a) \odot q$ is in g.
- $(d \gg (a \odot r)) \odot q$, for r in g_2: rewrites with (R19) to $d \gg (a \odot (r \odot q))$. s_2 is a subset of g_2 and $d \gg (a \odot (g_2 \odot g_2))$ is in g, so $d \gg (a \odot (r \odot q))$ is in g.
- $((d \gg a) \odot r) \odot q$, for r in g_2: rewrites with (R18) to $(d \gg a) \odot (r \odot q)$. s_2 is a subset of g_2 and $(d \gg a) \odot (g_2 \odot g_2))$ is in g, so $(d \gg a) \odot (r \odot q)$ is in g.
- $(d \gg (c \rhd r)) \odot q$, for r in g_2: rewrites with (R16) to $d \gg (c \rhd (r \odot q))$. s_2 is a subset of g_2 and $d \gg (c \rhd (g_2 \odot g_2))$ is in g, so $d \gg (c \rhd (r \odot q))$ is in g.

$\boxtimes$

Lemma 12 (Resulting form) *After applying guard to the right-hand sides of all equations in E, the right-hand sides of all equations in E are of the form g:*

$$\begin{array}{lll} g & ::= & g \oplus g \mid \delta \mid a \mid a \odot g_2 \mid c \rhd g_2 \mid d \gg a \mid \\ & & d \gg (a \odot g_2) \mid (d \gg a) \odot g_2 \mid d \gg (c \rhd g_2) \\ g_2 & ::= & \delta \mid a \mid X \mid g_2 \odot g_2 \mid d \gg a \mid d \gg X \end{array}$$

Proof We prove that $guard(p)$ has the form g, for any p of the form s. Lemma 8 states that the *guard* function is well-defined, so we can apply induction on the form s:

- a, δ, $c \rhd X$: trivial.
- X: $guard(X) = guard(rhs(X))$. The form of $rhs(X)$ is s, so our induction hypothesis applies and we conclude that $guard(X)$ has the form g.
- $p \oplus q$, for any p, q of the form s: $guard(p \oplus q) = guard(p) \oplus guard(q)$. The induction hypothesis applies to both operands. Therefore, $guard(p)$ and $guard(q)$ are both of the form g and $g \oplus g$ is in g.
- $d \gg p$, for any p of the form s_2: $guard(d \gg p) = rewr(d \gg guard(p))$. Since s_2 is a subset of s, p is also of the form s. Hence, the induction hypothesis applies, so $guard(p)$ is of the form g. By lemma 10, $rewr(d \gg guard(p))$ has the form g.
- $p \odot q$, for any p, q of the form s_2: $guard(p \odot q) = rewr(guard(p) \odot q)$. Since s_2 is a subset of s, p is also of the form s. Hence, the induction hypothesis applies, so $guard(p)$ is of the form g. By lemma 11, $rewr(guard(p) \odot q)$ has the form g.

All existing equations in E are rewritten into the form g and no new equations are added to E. Therefore, after applying *guard* to the right-hand sides of all equations in E, the right-hand sides of all equations in E are of the form g. $\boxtimes$

Lemma 13 (Soundness) *guard is sound.*

Proof Most rewrite rules are directed versions of HyPA axioms, except for rules (R12), (R15), (R19) and (R20). These rules are derivable from the axioms as follows:

$$\begin{array}{lll} (\text{R12}): & (c \rhd x) \odot y & \\ & \approx & ([\mathit{true}] \gg (c \rhd x)) \odot y \\ & \approx & [\mathit{true}] \gg c \rhd (x \odot y) \\ & \approx & c \rhd (x \odot y) \end{array} \qquad \begin{array}{lll} (\text{R15}): & \delta \odot x & \\ & \approx & ([\mathit{false}] \gg a) \odot x \\ & \approx & [\mathit{false}] \gg a \odot x \\ & \approx & \delta \end{array}$$

$$\begin{array}{lll} (\text{R19}): & (d \gg (a \odot x)) \odot y & \\ & \approx & ((d \gg a) \odot x) \odot y \\ & \approx & (d \gg a) \odot (x \odot y) \\ & \approx & d \gg (a \odot (x \odot y)) \end{array} \qquad \begin{array}{lll} (\text{R20}): & d \gg ((d' \gg a) \odot x) & \\ & \approx & d \gg (d' \gg (a \odot x)) \\ & \approx & (d \sim d') \gg (a \odot x) \end{array}$$

Therefore, the rewrite system *rewr* is sound. It is trivial to see that the *guard* function is sound, since it only substitutes recursion variables by their right-hand sides and the rewrite system is sound. ⊠

Post Processing The right-hand sides of the recursive equations are almost semi-linear now. In this fourth and final step, the equations only need to be cleaned up somewhat. This step consists of two consecutive transformation functions, P_1 and P_3. In P_1, new equations are introduced. This is done in a similar fashion as in the S_1 function and P_1 is not applied to these new equations. After this post processing step, all recursive equations in E are semi-linear. First, the function P_1 is applied to the right-hand sides of all recursive equations in E:

$$\begin{array}{lcl} P_1(p \oplus q) & = & P_1(p) \oplus P_1(q) \\ P_1(\delta) & = & \delta \\ P_1(a) & = & [\mathit{true}] \gg a \\ P_1(a \odot p) & = & [\mathit{true}] \gg (a \odot P_2(p)) \\ P_1(c \rhd p) & = & [\mathit{true}] \gg (c \rhd P_2(p)) \\ P_1(d \gg a) & = & d \gg a \\ P_1(d \gg (a \odot p)) & = & d \gg (a \odot P_2(p)) \\ P_1((d \gg a) \odot p) & = & d \gg (a \odot P_2(p)) \\ P_1(d \gg (c \rhd p)) & = & d \gg (c \rhd P_2(p)) \end{array}$$

$$\begin{array}{lcl} P_2(\delta) & = & Y, \text{ and } E := E \cup \{Y \approx \delta\} \\ P_2(a) & = & Y, \text{ and } E := E \cup \{Y \approx [\mathit{true}] \gg a\} \\ P_2(X) & = & X \\ P_2(p \odot q) & = & P_2(p) \odot P_2(q) \\ P_2(d \gg a) & = & Y, \text{ and } E := E \cup \{Y \approx d \gg a\} \\ P_2(d \gg X) & = & Y, \text{ and } E := E \cup \{Y \approx d \gg X\} \end{array}$$

Second, the function P_3 is applied to the right-hand sides of all recursive equations in E:

$$P_3(p) \quad = \quad \begin{cases} \mathit{rewr}(d \gg \mathit{rhs}(X)) & \text{if } p \text{ of the form } d \gg X \text{ for any re-initialization} \\ & \quad \text{clause } d \text{ and recursion variable } X \\ p & \text{otherwise} \end{cases}$$

where *rewr* is the rewrite system consisting of the following rules:

$$\begin{array}{lcll} d \gg (x \oplus y) & \to & d \gg x \oplus d \gg y & (\text{R11}) \\ d \gg \delta & \to & \delta & (\text{R14}) \\ d \gg (d' \gg x) & \to & (d \sim d') \gg x & (\text{R17}) \end{array}$$

The goal of these transformations is to make the right-hand sides of all equations in E semi-linear. Recall that a term is semi-linear if it is of the following form, for brevity called l here:

$$\begin{array}{lll} l & ::= & \delta \mid d \gg a \mid d \gg a \odot l_2 \mid d \gg c \rhd l_2 \mid l \oplus l \\ l_2 & ::= & X \mid l_2 \odot l_2 \end{array}$$

Lemma 14 *$P_2(p)$ is of the form l_2, for any p of the form g_2.*

Proof We prove this lemma by induction on the structure of the form g_2, with the lemma as the induction hypothesis. For the base cases δ, a, X, $d \gg a$ and $d \gg X$ this is trivial, since all of them give a single recursion variable X, which is in the form l_2.

Inductive case $p \odot q$, where p and q are of the form g_2: applying the induction hypothesis twice we see that $P_2(p)$ and $P_2(q)$ are both of the form l_2. Since $l_2 \odot l_2$ is in l_2, we conclude that $P_2(p \odot q)$ is in l_2. ⊠

Lemma 15 *After applying P_1 to the right-hand sides of all equations in E, these right-hand sides are either of the form l or of the form $d \gg X$ where $rhs(X)$ is of the form l.*

Proof Before applying P_1, the right-hand sides of all equations in E have the form g. We have two proof obligations:

- $P_1(p)$ is of the form l for any p of the form g: we prove this by induction on the structure of the form g. For the base cases δ, a and $d \gg a$ this is trivial. Inductive cases:
  - $p \oplus q$, where p and q are of the form g: applying the induction hypothesis twice, we see that both $P_1(p)$ and $P_1(q)$ are of the form l. Since $l \oplus l$ is in l, $P_1(p \oplus q)$ is also in l.
  - $a \odot p$, where p is of the form g_2: applying lemma 14 gives us that $P_2(p)$ is of the form l_2. Since $[true] \gg (a \odot l_2)$ is in l, $P_1(a \odot p)$ is also in l.
  - $c \rhd p$, where p is of the form g_2: applying lemma 14 gives us that $P_2(p)$ is of the form l_2. Since $[true] \gg (c \rhd l_2)$ is in l, $P_1(c \rhd p)$ is also in l.
  - $d \gg (a \odot p)$, where p is of the form g_2: analogous to $P_1(a \odot p)$.
  - $(d \gg a) \odot p)$, where p is of the form g_2: analogous to $P_1(a \odot p)$.
  - $d \gg (c \rhd p)$, where p is of the form g_2: analogous to $P_1(c \rhd p)$.
- all newly introduced equations are of the form l or of the form $d \gg X$ where $rhs(X)$ is of the form l: equations with right-hand sides of the following form might be introduced:
  - δ, which is in l.
  - $[true] \gg a$, which is in l.
  - $d \gg a$, which is in l.
  - $d \gg X$: we know that X was already present in E before the transformation, so its right-hand side after transformation is of the form l as shown above.

We conclude that after applying P_1 to the right-hand sides of all equations in E, these right-hand sides are either of the form l or of the form $d \gg X$ where $rhs(X)$ is of the form l. ⊠

Lemma 16 *$rewr(d \gg p)$ is of the form l, for any p of the form l.*

Proof We prove this by induction on the structure of $d \gg p$, for p of the form l.

- $d \gg \delta$: rewrites using (R14) to δ, which is in l.
- $d \gg (d \gg a)$: rewrites using (R17) to $(d \sim d) \gg a$, which is in l.
- $d \gg (d \gg (a \odot l_2))$: rewrites using (R17) to $(d \sim d) \gg a \odot l_2$, which is in l.
- $d \gg (d \gg (c \rhd l_2))$: rewrites using (R17) to $(d \sim d) \gg c \rhd l_2$, which is in l.
- $d \gg (p \oplus q)$: rewrites using (R11) to $rewr(d \gg p) \oplus rewr(d \gg q)$. Using the induction hypothesis, we see that both $rewr(d \gg p)$ and $rewr(d \gg q)$ are of the form l. Since $l \oplus l$ is in l, we conclude that $rewr(d \gg (p \oplus q))$ is in l.

$\boxtimes$

Lemma 17 (Resulting form) *After applying P_3 to the right-hand sides of all equations in E, the right-hand sides of all equations in E are semi-linear.*

Proof First, we prove that $P_3(p)$ is of the form l, for any p either of the form l or of the form $d \gg X$ where $rhs(X)$ is of the form l:

- p of the form $d \gg X$, where $rhs(X)$ is of the form l: $P_3(p) = P_3(d \gg X) = rewr(d \gg rhs(X))$. Lemma 16 states that $rewr(d \gg rhs(X))$ is of the form l, because $rhs(X)$ is of the form l. Therefore, $P_3(p)$ is of the form l.
- p of the form l: Since $d \gg X$ is not in the form l, $P_3(p) = p$. Therefore, $P_3(p)$ is of the form l.

We conclude that after applying P_3 to the right-hand sides of all equations in E, the right-hand sides of all equations in E are semi-linear. $\boxtimes$

Lemma 18 *$P_2(p) \approx p$ for any p of the form g_2.*

Proof We prove this by induction on the form g_2:

- δ: $P_2(\delta) = Y \approx \delta$.
- a: $P_2(a) = Y \approx [true] \gg a \approx a$.
- X: $P_2(X) = X$.
- $p \odot q$, where p and q are of the form g_2: applying the induction hypothesis twice, we see that $P_2(p \odot q) = P_2(p) \odot P_2(q) \approx p \odot q$.
- $d \gg a$: $P_2(d \gg a) = Y \approx d \gg a$.
- $d \gg X$: $P_2(d \gg X) = Y \approx d \gg X$.

$\boxtimes$

Lemma 19 (Soundness) *P_1 and P_3 are sound.*

Proof First, we prove that P_1 is sound, i.e. $P_1(p) \approx p$ for any p of the form g. We prove this by induction on the form g.

- $p \oplus q$, where p and q are of the form g: applying the induction hypothesis twice, we see that $P_1(p \oplus q) = P_1(p) \oplus P_1(q) \approx p \oplus q$.

- δ: $P_1(\delta) = \delta$.
- a: $P_1(a) = [true] \gg a \approx a$.
- $d \gg a$: $P_1(d \gg a) = d \gg a$.
- $a \odot p$, where p is of the form g_2: using lemma 18 we see that $P_1(a \odot p) = [true] \gg (a \odot P_2(p)) \approx [true] \gg (a \odot p) \approx a \odot p$.
- $c \rhd p$, where p is of the form g_2: analogous to $a \odot p$.
- $d \gg (a \odot p)$, where p is of the form g_2: analogous to $a \odot p$.
- $(d \gg a) \odot p$, where p is of the form g_2: analogous to $a \odot p$.
- $d \gg (c \rhd p)$, where p is of the form g_2: analogous to $a \odot p$.

The rewrite system used in P_3 is sound, because all rewrite rules are directed versions of HyPA axioms.

Second, we prove that P_3 is sound, i.e. $P_3(p) \approx p$ for any p. We have two cases:

- p of the form $d \gg X$: Since *rewr* is sound, $P_3(p) = P_3(d \gg X) = rewr(d \gg rhs(X)) \approx d \gg rhs(X) \approx d \gg X$.
- p not of the form $d \gg X$: $P_3(p) = p$.

We conclude that P_1 and P_3 are both sound. ⊠

Lemma 20 (Well-definedness) *P_1 and P_3 are well-defined.*

Proof The rewrite system used in this step is a subset of the rewrite system used in the *guard* function. Since the rewrite system of the *guard* function is terminating, the rewrite system used in this step is terminating as well.

Well-definedness of P_1 and P_3 is trivial, since P_1 and P_3 do not occur in the right-hand sides of their definitions. ⊠

4.3.2 Stage 2: From Semi-linear to Abstracted-linear

In this stage, the semi-linear equations are transformed to a form where only a single recursion variable occurs after an action or a flow clause. This transformation introduces a stack of (a representation of) recursion variables. The idea is that this stack is a kind of to-do list. When a sequential composition of multiple recursion variables (i.e. $X_0 \odot \ldots \odot X_n$) is encountered, they are all pushed onto the stack and the process modeled by the recursion variable on top of the stack starts executing (i.e. X_0). As soon as this process terminates, execution is resumed with the next process on the stack (if any). This stack is represented by a fresh model variable. This variable is abstracted from, because it should not be visible externally.

Currently, the right-hand sides of all recursive equations are in semi-linear form. Another way to denote this form is the following:

$$\begin{array}{rcll} X_i & \approx & \bigoplus_{j \in J(i)} & d_j \gg a_j \oplus \\ & & \bigoplus_{k \in K(i)} & d_k \gg (a_k \odot (X_{f(k,1)} \odot \ldots \odot X_{f(k,n)})) \oplus \\ & & \bigoplus_{l \in L(i)} & d_l \gg (c_l \rhd (X_{f(l,1)} \odot \ldots \odot X_{f(l,n')})) \end{array}$$

The $\bigoplus$-notation is used as a shorthand for the alternative composition of a finite number of terms. If the domain of this quantifier is empty, it is deadlock (δ). The index i ranges from 1 to

the number of recursive equations in the specification. The sets $J(i)$, $K(i)$ and $L(i)$ are disjoint for all such i.

The transformation goes as follows. First, a single new recursive equation A is defined, which captures the contents of *all* equations in E. A is defined as follows (note that A is linear):

$$
\begin{aligned}
A \;\approx\; & \bigoplus_{X_i \in Var(E)} \\
& \Bigg(\bigoplus_{j \in J(i)} \left[s^- \neq \emptyset \;\wedge\; get(s^-) = X_i \;\wedge\; pop(s^-) = \emptyset \right] \sim d_j \gg a_j \\
& \oplus \\
& \bigoplus_{j \in J(i)} \left[s \,\middle|\, s^- \neq \emptyset \;\wedge\; get(s^-) = X_i \;\wedge\; pop(s^-) \neq \emptyset \;\wedge\; s^+ = pop(s^-) \right] \sim d_j \gg a_j \odot A \\
& \oplus \\
& \bigoplus_{k \in K(i)} \left[\; s \;\middle|\; \begin{array}{l} s^- \neq \emptyset \;\wedge\; get(s^-) = X_i \wedge \\ s^+ = push(X_{f(k,1)}, \ldots push(X_{f(k,n_k)}, pop(s^-)) \cdots) \end{array} \right] \sim d_k \gg a_k \odot A \\
& \oplus \\
& \bigoplus_{l \in L(i)} \left[\; s \;\middle|\; \begin{array}{l} s^- \neq \emptyset \;\wedge\; get(s^-) = X_i \;\wedge\; s^+ = push(X_{f(l,1)}, \\ \ldots push(X_{f(l,n_l)}, pop(s^-)) \cdots) \end{array} \right] \sim d_l \gg \left(\begin{array}{l} c_l \wedge \\ (s | \, \dot{s} = 0) \end{array} \right) \rhd A \Bigg)
\end{aligned}
$$

where the operations on a stack variable s are defined as follows:

$\emptyset$	Represents the empty stack.
$push(i, s)$	Returns the stack s with element i pushed onto it.
$pop(s)$	Returns the stack s without the top element.
$get(s)$	Returns the top element of the stack s (but doesn't pop it).

Second, all recursive equations in E are transformed, except for the newly introduced equation A, as follows:

$$\text{For all } X_i \in E \backslash \{A\},\; X_i \approx \left[\!\left[\; s \;\middle|\; \left[s \,|\, s^+ = push(X_i, \emptyset) \right] \gg A \; \right]\!\right]$$

Because of the abstraction, the right-hand sides of the equations are not transformed into linear form, but into a form called *abstracted-linear*. A term is in abstracted-linear form if it has the form $[\![V \mid d \gg X]\!]$, and the right-hand side of the variable X is linear. The result of the transformation in this stage is that the right-hand sides of all recursive equations in E are abstracted-linear, except for the single linear recursive equation A.

Lemma 21 *For any $s \in \mathcal{V}_{\mathrm{m}}$, re-initialization clause d and constant expressions C, C', where $s \notin Var(d)$ and $\equiv$ denotes equivalence of re-initialization clauses:*

$$
\begin{array}{lcll}
[s \,|\, s^+ = C] \sim [s \,|\, s^+ = C'] & \equiv & [s \,|\, s^+ = C'] & (1) \\
d \sim [s \,|\, s^+ = C] & \equiv & [s \,|\, s^+ = C] \sim d & (2) \\
[s \,|\, s^+ = C] & \equiv & [s \,|\, true] \sim [s \,|\, s^+ = C] & (3) \\
[s \,|\, pop(s^-) \neq \emptyset \;\wedge\; s^+ = C] & \equiv & [pop(s^-) \neq \emptyset] \sim [s \,|\, s^+ = C] & (4)
\end{array}
$$

Proof See appendix B.1 for the proofs. ⊠

Lemma 22 *For any $s \in \mathcal{V}_{\mathrm{m}}$, $a \in \mathcal{A}$, $x \in \mathcal{T}$, re-initialization clauses d, d_s and flow clause c, where $s \notin Var(c) \cup Var(x)$ and $d_s = [s\,|s^+ = C\,]$ with C a constant expression:*

$$
\begin{array}{lcll}
[\![\ s\ \mid d_s \gg x\ ]\!] & \approx & x & (1)\\
[\![\ s\ \mid (d \sim d_s) \gg a \odot x\ ]\!] & \approx & [\![\ s\ \mid d \gg a \odot (d_s \gg x)\ ]\!] & (2)\\
[\![\ s\ \mid (d \sim d_s) \gg (c \wedge (\,s|\,\dot{s} = 0)) \rhd x\ ]\!] & \approx & [\![\ s\ \mid d \gg c \rhd (d_s \gg x)\ ]\!] & (3)
\end{array}
$$

Proof See appendix B.2 for the proofs. ⊠

Lemma 23 (Soundness) *The transformation in this stage is sound.*

Proof Using the principle RSP, we prove that every $X_i \in E\backslash\{A\}$ before transformation is equivalent to $[\![\ s\ \mid [s\,|s^+ = push(X_i, \emptyset)] \gg A\ ]\!]$. We first present some derivations, which are used later on in the proof.

For any $w_0, \ldots, w_n$, for $n \geq 0$ and where for all w_i, $X_{w_i} \in Var(E)\backslash\{A\}$:

$[\![\ s\ \mid [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \gg A\ ]\!]$

$\approx$ { Substitute A by its right-hand side, distribute $\gg$}

$$
\begin{array}{l}
|[\ s \mid \bigoplus_{X_i \in Var(E)} (\\
\bigoplus_{j \in J(i)} \left(\begin{array}{l} [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim \\ [s^- \neq \emptyset \wedge get(s^-) = X_i \wedge pop(s^-) = \emptyset] \sim d_j \end{array} \right) \gg a_j \oplus\\
\bigoplus_{j \in J(i)} \left(\begin{array}{l} [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim \\ \left[\ s \ \middle|\ \begin{array}{l} s^- \neq \emptyset \wedge get(s^-) = X_i \wedge \\ pop(s^-) \neq \emptyset \wedge s^+ = pop(s^-) \end{array} \right] \sim d_j \end{array} \right) \gg a_j \odot A \oplus\\
\bigoplus_{k \in K(i)} \left(\begin{array}{l} [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim \\ \left[\ s \ \middle|\ \begin{array}{l} s^- \neq \emptyset \wedge get(s^-) = X_i \wedge \\ s^+ = push(X_{f(k,1)}, \ldots push(X_{f(k,n_k)}, pop(s^-))\cdots) \end{array} \right] \sim d_k \end{array} \right) \gg a_k \odot A \oplus\\
\bigoplus_{l \in L(i)} \left(\begin{array}{l} [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim \\ \left[\ s \ \middle|\ \begin{array}{l} s^- \neq \emptyset \wedge get(s^-) = X_i \wedge \\ s^+ = push(X_{f(l,1)}, \ldots \\ push(X_{f(l,n_l)}, pop(s^-))\cdots) \end{array} \right] \sim d_l \end{array} \right) \gg (c_l \wedge (\,s|\,\dot{s} = 0)) \rhd A\\
)\ ]|
\end{array}
$$

$\approx$ { Eliminate sum: all summands are δ except for summands of X_{w_0}}

$$
\begin{array}{l}
|[\ s \mid\\
\bigoplus_{j \in J(w_0)} \left(\begin{array}{l} [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim \\ [s^- \neq \emptyset \wedge get(s^-) = X_{w_0} \wedge pop(s^-) = \emptyset] \sim d_j \end{array} \right) \gg a_j \oplus\\
\bigoplus_{j \in J(w_0)} \left(\begin{array}{l} [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim \\ \left[\ s \ \middle|\ \begin{array}{l} s^- \neq \emptyset \wedge get(s^-) = X_{w_0} \wedge \\ pop(s^-) \neq \emptyset \wedge s^+ = pop(s^-) \end{array} \right] \sim d_j \end{array} \right) \gg a_j \odot A \oplus\\
\bigoplus_{k \in K(w_0)} \left(\begin{array}{l} [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim \\ \left[\ s \ \middle|\ \begin{array}{l} s^- \neq \emptyset \wedge get(s^-) = X_{w_0} \wedge \\ s^+ = push(X_{f(k,1)}, \ldots push(X_{f(k,n_k)}, pop(s^-))\cdots) \end{array} \right] \sim d_k \end{array} \right) \gg a_k \odot A \oplus\\
\bigoplus_{l \in L(w_0)} \left(\begin{array}{l} [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \\ \sim \left[\ s \ \middle|\ \begin{array}{l} s^- \neq \emptyset \wedge get(s^-) = X_{w_0} \wedge \\ s^+ = push(X_{f(l,1)}, \ldots \\ push(X_{f(l,n_l)}, pop(s^-))\cdots) \end{array} \right] \sim d_l \end{array} \right) \gg (c_l \wedge (\,s|\,\dot{s} = 0)) \rhd A\\
]|
\end{array}
$$

$\approx$ { Calculation on re-initialization clauses }

$$
\begin{array}{l}
|[\ s\ | \\
\bigoplus_{j\in J(w_0)} ([s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim [pop(s^-) = \emptyset] \sim d_j) \gg a_j \ \oplus \\
\bigoplus_{j\in J(w_0)} \left(\begin{array}{l} [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim \\ [s\,|pop(s^-) \neq \emptyset \wedge s^+ = push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim d_j \end{array} \right) \gg a_j \odot A \oplus \\
\bigoplus_{k\in K(w_0)} \left(\begin{array}{l} [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim \\ \left[\ s \ \middle|\ \begin{array}{l} s^+ = push(X_{f(k,1)}, \ldots \\ push(X_{f(k,n_k)}, push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots))\cdots) \end{array} \right] \sim d_k \end{array} \right) \gg a_k \odot A \oplus \\
\bigoplus_{l\in L(w_0)} \left(\begin{array}{l} [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \\ \sim \left[\ s \ \middle|\ \begin{array}{l} s^+ = push(X_{f(l,1)}, \ldots \\ push(X_{f(l,n_l)}, push(X_{w_1}, \ldots \\ push(X_{w_n}, \emptyset)\cdots))\cdots) \end{array} \right] \sim d_l \end{array} \right) \gg (c_l \wedge (\,s|\,\dot{s} = 0)) \rhd A \\
]|
\end{array}
$$

$\approx$ { Lemma 21 (1) and (4) }

$$
\begin{array}{l}
|[\ s\ | \\
\bigoplus_{j\in J(w_0)} ([s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim [pop(s^-) = \emptyset] \sim d_j) \gg a_j \ \oplus \\
\bigoplus_{j\in J(w_0)} \left(\begin{array}{l} [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim [pop(s^-) \neq \emptyset] \sim \\ [s\,|s^+ = push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim d_j \end{array} \right) \gg a_j \odot A \oplus \\
\bigoplus_{k\in K(w_0)} \left[\ s \ \middle|\ \begin{array}{l} s^+ = push(X_{f(k,1)}, \ldots \\ push(X_{f(k,n_k)}, push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots))\cdots) \end{array} \right] \sim d_k \gg a_k \odot A \oplus \\
\bigoplus_{l\in L(w_0)} \left[\ s \ \middle|\ \begin{array}{l} s^+ = push(X_{f(l,1)}, \ldots \\ push(X_{f(l,n_l)}, push(X_{w_1}, \ldots \\ push(X_{w_n}, \emptyset)\cdots))\cdots) \end{array} \right] \sim d_l \gg (c_l \wedge (\,s|\,\dot{s} = 0)) \rhd A \\
]|
\end{array}
$$

$\approx$ { Distribute abstraction }

$$
\begin{array}{l}
\bigoplus_{j\in J(w_0)} [\![\ s\ |\ ([s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim [pop(s^-) = \emptyset] \sim d_j) \gg a_j\ ]\!] \oplus \\
\bigoplus_{j\in J(w_0)} \left\| \left[\ s \ \middle|\ \left(\begin{array}{l} [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim [pop(s^-) \neq \emptyset] \sim \\ [s\,|s^+ = push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim d_j \end{array} \right) \gg a_j \odot A \right] \right\| \oplus \\
\bigoplus_{k\in K(w_0)} \left\| \left[\ s \ \middle|\ \left[\ s \ \middle|\ \begin{array}{l} s^+ = push(X_{f(k,1)}, \ldots push(X_{f(k,n_k)}, \\ push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots))\cdots) \end{array} \right] \sim d_k \gg a_k \odot A \right] \right\| \oplus \\
\bigoplus_{l\in L(w_0)} \left\| \left[\ s \ \middle|\ \left[\ s \ \middle|\ \begin{array}{l} s^+ = push(X_{f(l,1)}, \ldots \\ push(X_{f(l,n_l)}, push(X_{w_1}, \ldots \\ push(X_{w_n}, \emptyset)\cdots))\cdots) \end{array} \right] \sim d_l \gg (c_l \wedge (\,s|\,\dot{s} = 0)) \rhd A \right] \right\|
\end{array}
$$

$\approx$ { Lemma 21 (2) }

$$
\begin{array}{l}
\bigoplus_{j\in J(w_0)} [\![\ s\ |\ ([s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim [pop(s^-) = \emptyset] \sim d_j) \gg a_j\ ]\!] \oplus \\
\bigoplus_{j\in J(w_0)} \left\| \left[\ s \ \middle|\ \left(\begin{array}{l} [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim [pop(s^-) \neq \emptyset] \\ \sim d_j \sim [s\,|s^+ = push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots)] \end{array} \right) \gg a_j \odot A \right] \right\| \oplus \\
\bigoplus_{k\in K(w_0)} \left\| \left[\ s \ \middle|\ d_k \sim \left[\ s \ \middle|\ \begin{array}{l} s^+ = push(X_{f(k,1)}, \ldots push(X_{f(k,n_k)}, \\ push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots))\cdots) \end{array} \right] \gg a_k \odot A \right] \right\| \oplus \\
\bigoplus_{l\in L(w_0)} \left\| \left[\ s \ \middle|\ d_l \sim \left[\ s \ \middle|\ \begin{array}{l} s^+ = push(X_{f(l,1)}, \ldots \\ push(X_{f(l,n_l)}, push(X_{w_1}, \ldots \\ push(X_{w_n}, \emptyset)\cdots))\cdots) \end{array} \right] \gg (c_l \wedge (\,s|\,\dot{s} = 0)) \rhd A \right] \right\|
\end{array}
$$

$\approx$ { Lemma 22 (2) and (3) }

$$\bigoplus_{j\in J(w_0)} [\![\ s\ \mid ([s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim [pop(s^-) = \emptyset] \sim d_j) \gg a_j\ ]\!] \oplus$$
$$\bigoplus_{j\in J(w_0)} \left|\!\left[s \;\middle|\; \begin{array}{l} \left(\begin{array}{l} [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \\ \sim [pop(s^-) \neq \emptyset] \sim d_j \end{array} \right) \gg a_j \\ \odot\, [s\,|s^+ = push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots)] \gg A \end{array} \right]\!\right| \oplus$$
$$\bigoplus_{k\in K(w_0)} \left|\!\left[s \;\middle|\; d_k \gg a_k \odot \left[s \;\middle|\; \begin{array}{l} s^+ = push(X_{f(k,1)}, \ldots \\ push(X_{f(k,n_k)}, push(X_{w_1}, \\ \ldots push(X_{w_n}, \emptyset)\cdots)\cdots) \end{array} \right] \gg A \right]\!\right| \oplus$$
$$\bigoplus_{l\in L(w_0)} \left|\!\left[s \;\middle|\; d_l \gg c_l \rhd \left[s \;\middle|\; \begin{array}{l} s^+ = push(X_{f(l,1)}, \ldots push(X_{f(l,n_l)}, \\ push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots)\cdots) \end{array} \right] \gg A \right]\!\right|$$

$\approx$ { Lemma 21 (3) }

$$\bigoplus_{j\in J(w_0)} \left|\!\left[\ s\ \middle|\ \left(\ [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim [pop(s^-) = \emptyset] \sim d_j\ \right) \gg a_j\ \right]\!\right| \oplus$$
$$\bigoplus_{j\in J(w_0)} \left|\!\left[s \;\middle|\; \begin{array}{l} \left(\begin{array}{l} [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim \\ [pop(s^-) \neq \emptyset] \sim d_j \end{array} \right) \gg a_j \odot \\ [s\,|true] \gg [s\,|s^+ = push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots)] \gg A \end{array} \right]\!\right| \oplus$$
$$\bigoplus_{k\in K(w_0)} \left|\!\left[s \;\middle|\; \begin{array}{l} d_k \gg a_k \odot [s\,|true] \gg \\ \left[s \;\middle|\; \begin{array}{l} s^+ = push(X_{f(k,1)}, \ldots push(X_{f(k,n_k)}, \\ push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots)\cdots) \end{array} \right] \gg A \end{array} \right]\!\right| \oplus$$
$$\bigoplus_{l\in L(w_0)} \left|\!\left[s \;\middle|\; \begin{array}{l} d_l \gg c_l \rhd [s\,|true] \gg \\ \left[s \;\middle|\; \begin{array}{l} s^+ = push(X_{f(l,1)}, \ldots push(X_{f(l,n_l)}, \\ push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots)\cdots) \end{array} \right] \gg A \end{array} \right]\!\right|$$

$\approx$ { Distribution of abstraction (VA2, VA3) }

$$\bigoplus_{j\in J(w_0)} [\![\ s\ \mid ([s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim [pop(s^-) = \emptyset] \sim d_j) \gg a_j\ ]\!] \oplus$$
$$\bigoplus_{j\in J(w_0)} [\![\ s\ \mid ([s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim [pop(s^-) \neq \emptyset] \sim d_j) \gg a_j\ ]\!] \odot$$
$$[\![\ s\ \mid [s\,|s^+ = push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots)] \gg A\ ]\!] \oplus$$
$$\bigoplus_{k\in K(w_0)} [\![\ s\ \mid d_k \gg a_k\ ]\!] \odot \left|\!\left[s \;\middle|\; \left[s \;\middle|\; \begin{array}{l} s^+ = push(X_{f(k,1)}, \ldots push(X_{f(k,n_k)}, \\ push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots)\cdots) \end{array} \right] \gg A \right]\!\right| \oplus$$
$$\bigoplus_{l\in L(w_0)} [\![\ s\ \mid d_l \gg c_l\ ]\!] \rhd \left|\!\left[s \;\middle|\; \left[s \;\middle|\; \begin{array}{l} s^+ = push(X_{f(l,1)}, \ldots push(X_{f(l,n_l)}, \\ push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots)\cdots) \end{array} \right] \gg A \right]\!\right|$$

$\approx$ { Case distinction on n}

If $n = 0$:

$$\bigoplus_{j\in J(w_0)} [\![\ s\ \mid [s\,|s^+ = push(X_{w_0}, \emptyset)] \sim d_j \gg a_j\ ]\!] \oplus$$
$$\bigoplus_{k\in K(w_0)} [\![\ s\ \mid d_k \gg a_k\ ]\!] \odot \left|\!\left[\ s\ \middle|\ \left[s\,|s^+ = push(X_{f(k,1)}, \ldots push(X_{f(k,n_k)}, \emptyset)\cdots)\right] \gg A\ \right]\!\right| \oplus$$
$$\bigoplus_{l\in L(w_0)} [\![\ s\ \mid d_l \gg c_l\ ]\!] \rhd \left|\!\left[\ s\ \middle|\ \left[s\,|s^+ = push(X_{f(l,1)}, \ldots push(X_{f(l,n_l)}, \emptyset)\cdots)\right] \gg A\ \right]\!\right|$$

If $n > 0$:

$$\bigoplus_{j\in J(w_0)} [\![\ s\ \mid [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \sim d_j \gg a_j\ ]\!] \odot$$
$$[\![\ s\ \mid [s\,|s^+ = push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots)] \gg A\ ]\!] \oplus$$
$$\bigoplus_{k\in K(w_0)} [\![\ s\ \mid d_k \gg a_k\ ]\!] \odot \left|\!\left[s \;\middle|\; \left[s \;\middle|\; \begin{array}{l} s^+ = push(X_{f(k,1)}, \ldots push(X_{f(k,n_k)}, \\ push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots)\cdots) \end{array} \right] \gg A \right]\!\right| \oplus$$
$$\bigoplus_{l\in L(w_0)} [\![\ s\ \mid d_l \gg c_l\ ]\!] \rhd \left|\!\left[s \;\middle|\; \left[s \;\middle|\; \begin{array}{l} s^+ = push(X_{f(l,1)}, \ldots push(X_{f(l,n_l)}, \\ push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots)\cdots) \end{array} \right] \gg A \right]\!\right|$$

$\approx$ { Lemma 22 (1) and the axiom $[\![\ V\ \mid x\ ]\!] \approx x$, if $Var(x) \cap V = \emptyset$}

If $n = 0$:

$$\bigoplus_{j\in J(w_0)} d_j \gg a_j \ \oplus$$
$$\bigoplus_{k\in K(w_0)} d_k \gg a_k \ \odot\ \left\|\left[\!\left[\ s\ \middle|\ \left[s\,\middle|s^+ = push(X_{f(k,1)}, \ldots push(X_{f(k,n_k)}, \emptyset)\cdots)\right] \gg A\ \right]\!\right\| \oplus$$
$$\bigoplus_{l\in L(w_0)} d_l \gg c_l \ \rhd\ \left\|\left[\!\left[\ s\ \middle|\ \left[s\,\middle|s^+ = push(X_{f(l,1)}, \ldots push(X_{f(l,n_l)}, \emptyset)\cdots)\right] \gg A\ \right]\!\right\|$$

If $n > 0$:

$$\bigoplus_{j\in J(w_0)} d_j \gg a_j \ \odot\ [\![\ s\ \mid [s\,|s^+ = push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots)] \gg A\ ]\!] \oplus$$
$$\bigoplus_{k\in K(w_0)} d_k \gg a_k \ \odot\ \left\|\left[\!\left[\ s\ \middle|\ \left[\!\left[\ s\ \middle|\ \begin{array}{l} s^+ = push(X_{f(k,1)}, \ldots push(X_{f(k,n_k)}, \\ push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots))\cdots) \end{array}\right]\!\right] \gg A\ \right]\!\right\| \oplus$$
$$\bigoplus_{l\in L(w_0)} d_l \gg c_l \ \rhd\ \left\|\left[\!\left[\ s\ \middle|\ \left[\!\left[\ s\ \middle|\ \begin{array}{l} s^+ = push(X_{f(l,1)}, \ldots push(X_{f(l,n_l)}, \\ push(X_{w_1}, \ldots push(X_{w_n}, \emptyset)\cdots))\cdots) \end{array}\right]\!\right] \gg A\ \right]\!\right\|$$

$$X_{w_0} \odot \ldots \odot X_{w_n}$$
$$\approx \left(\begin{array}{l} \bigoplus_{j\in J(w_0)} d_j \gg a_j \oplus \\ \bigoplus_{k\in K(w_0)} d_k \gg a_k \odot X_{f(k,1)} \odot \ldots \odot X_{f(k,n_k)} \oplus \\ \bigoplus_{l\in L(w_0)} d_l \gg c_l \rhd X_{f(l,1)} \odot \ldots \odot X_{f(l,n_l)} \end{array}\right) \odot X_{w_1} \odot \ldots \odot X_{w_n}$$
$$\approx \begin{cases} \bigoplus_{j\in J(w_0)} d_j \gg a_j \oplus \\ \bigoplus_{k\in K(w_0)} d_k \gg a_k \odot X_{f(k,1)} \odot \ldots \odot X_{f(k,n_k)} \oplus & \text{if } n = 0 \\ \bigoplus_{l\in L(w_0)} d_l \gg c_l \rhd X_{f(l,1)} \odot \ldots \odot X_{f(l,n_l)} \\ \\ \bigoplus_{j\in J(w_0)} d_j \gg a_j \odot X_{w_1} \odot \ldots \odot X_{w_n} \oplus \\ \bigoplus_{k\in K(w_0)} d_k \gg a_k \odot X_{f(k,1)} \odot \ldots \odot X_{f(k,n_k)} \odot X_{w_1} \odot \ldots \odot X_{w_n} \oplus & \text{if } n > 0 \\ \bigoplus_{l\in L(w_0)} d_l \gg c_l \rhd X_{f(l,1)} \odot \ldots \odot X_{f(l,n_l)} \odot X_{w_1} \odot \ldots \odot X_{w_n} \end{cases}$$

We introduce an (infinite) set of recursive equations $Y_{w_0\ldots w_n}$ for every sequence $w_0 \ldots w_n$. These equations are defined as follows:

$$Y_{w_0\ldots w_n} \approx \begin{cases} \bigoplus_{j\in J(w_0)} d_j \gg a_j \oplus \\ \bigoplus_{k\in K(w_0)} d_k \gg a_k \odot Y_{w_{f(k,1)}\ldots w_{f(k,n_k)}} \oplus & \text{if } n = 0 \\ \bigoplus_{l\in L(w_0)} d_l \gg c_l \rhd Y_{w_{f(l,1)}\ldots w_{f(l,n_l)}} \\ \\ \bigoplus_{j\in J(w_0)} d_j \gg a_j \odot Y_{w_1\ldots w_n} \oplus \\ \bigoplus_{k\in K(w_0)} d_k \gg a_k \odot Y_{w_{f(k,1)}\ldots w_{f(k,n_k)} w_1\ldots w_n} \oplus & \text{if } n > 0 \\ \bigoplus_{l\in L(w_0)} d_l \gg c_l \rhd Y_{w_{f(l,1)}\ldots w_{f(l,n_l)} w_1\ldots w_n} \end{cases}$$

We see that $[\![\ s\ \mid [s\,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset)\cdots)] \gg A\ ]\!]$ as well as $X_{w_0} \odot \ldots \odot X_{w_n}$

are solutions of $Y_{w_0 \ldots w_n}$. Since all these equations are guarded, we can apply RSP, so we have that

$$X_{w_0} \odot \ldots \odot X_{w_n} \approx \left[\!\left[s \mid \left[s \,|s^+ = push(X_{w_0}, \ldots push(X_{w_n}, \emptyset) \cdots) \right] \gg A \right]\!\right].$$

From this we conclude that it is sound to transform the right-hand sides of all equations $X_i \in E \backslash \{A\}$ into $[\![s \mid [s \,|s^+ = push(X_i, \emptyset)] \gg A]\!]$, so the transformation in this stage is sound. ⊠

Lemma 24 (Well-definedness) *The transformation in this stage is well-defined.*

Proof Trivial. ⊠

Lemma 25 (Resulting form) *After applying the transformation in this stage, the right-hand sides of all equations in E are abstracted-linear, except one equation, whose right-hand side is linear.*

Proof It is straightforward to verify that the recursive equation A is linear. Furthermore, it is straightforward to see that all other recursive equations in E are transformed into abstracted-linear equations. ⊠

4.3.3 Stage 3: Transforming the Initial Term

In this final stage, the initial term t is transformed to the form $[\![V \mid d \gg X]\!]$, where X is linear. The transformation is based on two operations. The first operation computes the parallel composition of two recursion variables whose right-hand side is abstracted-linear. The result is a single recursion variable whose right-hand side is also abstracted-linear. The second operation computes the encapsulation of a variable whose right-hand side is abstracted-linear and the result is another single recursion variable whose right-hand side is abstracted-linear as well. The transformation function T_1 below uses these two operations to eliminate all parallel composition and encapsulation operators from t, so the initial term becomes an abstraction of a single variable whose right-hand side is abstracted-linear.

The function T_1 is applied to the initial term t and is defined as follows:

$$T_1(p) = \begin{cases} [\![V \mid T_2(q)]\!] & \text{if } p \text{ is of the form } [\![V \mid q]\!] \\ [\![\emptyset \mid T_2(p)]\!] & \text{otherwise} \end{cases}$$

$$\begin{aligned} T_2(X) &= X \\ T_2(p \parallel q) &= elim\_par\_comp(T_2(p), T_2(q)) \\ T_2(\partial_H(p)) &= elim\_encap(H, T_2(p)) \end{aligned}$$

The result of $T_2(p)$ is always a single recursion variable, say X, whose right-hand side is abstracted-linear. Therefore, the result of $T_1(p)$ is always an abstraction of X, i.e. $[\![V \mid X]\!]$.

To obtain the final linear recursive specification, this final transformation is performed on the initial term. Suppose $X \approx [\![W \mid d \gg A]\!]$, where A is linear, then:

$$t = [\![V \mid X]\!] \approx [\![V \mid [\![W \mid d \gg A]\!]]\!] \approx [\![V \cup W \mid d \gg A]\!]$$

The specification $\langle\, t \,|\, \{A\} \,\rangle$ is taken as the resulting specification. This resulting specification is linear, so the linearization algorithm is finished.

First, the operations for the elimination of parallel composition and encapsulation are introduced and the corresponding correctness proofs are given. Then, the correctness of T_1 is considered.

Eliminating Parallel Composition The *elim_par_comp* function takes two recursion variables whose right-hand sides are abstracted-linear, and it returns a single recursion variable whose right-hand side is abstracted-linear as well. The returned recursion variable is equivalent to the parallel composition of the two arguments.

Suppose X and Y are abstracted-linear, and the parallel composition in $X \parallel Y$ needs to be eliminated. From the previous stage it is known that all recursive equations in E are abstracted-linear, except for the single linear recursive equation A. The functions T_1, *elim_par_comp* and *elim_encap* maintain this property, so X and Y both refer to the same linear equation A. Therefore, X and Y have the following form:

$$\begin{array}{lcl} X & \approx & [\![\, S \mid d_X \gg A \,]\!] \\ Y & \approx & [\![\, T \mid d_Y \gg A \,]\!] \\ A & \approx & \bigoplus_{j \in J(A)} d_j \gg a_j \oplus \bigoplus_{k \in K(A)} d_k \gg a_k \odot A \oplus \bigoplus_{l \in L(A)} d_l \gg c_l \rhd A \end{array}$$

All equations in the set of equations E of the specification at this point are abstracted-linear, except for the linear equation for A. We denote a linear term that contains only the variable A as $lin(A)$ and we denote an abstracted-linear term that contains only the variable A as $abslin(A)$. Suppose $P_0, \ldots, P_n$ are all the recursion variables defined in E except A, X and Y. Then, before eliminating parallel composition:

$$E = \left\{ \begin{array}{l} A \approx lin(A), X \approx abslin(A), Y \approx abslin(A), \\ P_0 \approx abslin(A), \ldots, P_n \approx abslin(A) \end{array} \right\}$$

The following steps are performed in the order they are listed:

1. First, by applying the renaming axiom for abstraction, all variables in T that are in both S and T are renamed, in order to make S and T disjoint. This is done by applying axioms (VA5) and (VA9) multiple times and it leads to a fresh equation, say B, in which these variables are renamed. Y then refers to B instead of A. B is added to E.

$$\begin{array}{lcl} X & \approx & [\![\, S \mid d_X \gg A \,]\!] \\ A & \approx & \bigoplus_{j \in J(A)} d_j \gg a_j \oplus \\ & & \bigoplus_{k \in K(A)} d_k \gg a_k \odot A \oplus \\ & & \bigoplus_{l \in L(A)} d_l \gg c_l \rhd A \end{array} \qquad \begin{array}{lcl} Y & \approx & [\![\, T \mid d_Y \gg B \,]\!] \\ B & \approx & \bigoplus_{j \in J(B)} d_j \gg a_j \oplus \\ & & \bigoplus_{k \in K(B)} d_k \gg a_k \odot B \oplus \\ & & \bigoplus_{l \in L(B)} d_l \gg c_l \rhd B \end{array}$$

   Now
$$E = \left\{ \begin{array}{l} A \approx lin(A), B \approx lin(B), X \approx abslin(A), Y \approx abslin(B), \\ P_0 \approx abslin(A), \ldots, P_n \approx abslin(A) \end{array} \right\}$$

2. Now, the parallel composition of X and Y is computed, which gives the abstracted-linear recursive equation Z and the linear equations Z_1 to $Z_{11,l}$. These new equations are *not* added to E.

$$Z \approx X \parallel Y$$

$$
\begin{aligned}
&\approx \quad [\![\ S \mid d_X \gg A\ ]\!] \parallel [\![\ T \mid d_Y \gg B\ ]\!] \\
&\approx \quad [\![\ S \mid d_X \gg A \parallel [\![\ T \mid d_Y \gg B\ ]\!]\ ]\!] \\
&\approx \quad [\![\ S \mid [\![\ T \mid d_X \gg A \parallel d_Y \gg B\ ]\!]\ ]\!] \\
&\approx \quad [\![\ S \cup T \mid d_X \gg A \parallel d_Y \gg B\ ]\!] \\
&\approx \quad [\![\ S \cup T \mid [true] \gg Z_1\ ]\!]
\end{aligned}
$$

$$
\begin{aligned}
Z_1 \quad &\approx \quad d_X \gg A \parallel d_Y \gg B \\
&\approx \quad \bigoplus_{j \in J(A)} (d_X \sim d_j) \gg a_j \odot Z_6 \oplus \\
&\qquad \bigoplus_{k \in K(A)} (d_X \sim d_k) \gg a_k \odot Z_2 \oplus \\
&\qquad \bigoplus_{j' \in J(B)} (d_Y \sim d_{j'}) \gg a_{j'} \odot Z_5 \oplus \\
&\qquad \bigoplus_{k' \in K(B)} (d_Y \sim d_{k'}) \gg a_{k'} \odot Z_3 \oplus \\
&\qquad \bigoplus_{j \in J(A)} \bigoplus_{j' \in J(B)} ((d_X \sim d_j) \wedge (d_Y \sim d_{j'})) \gg \gamma(a_j, a_{j'}) \oplus \\
&\qquad \bigoplus_{k \in K(A)} \bigoplus_{j' \in J(B)} ((d_X \sim d_k) \wedge (d_Y \sim d_{j'})) \gg \gamma(a_k, a_{j'}) \odot A \oplus \\
&\qquad \bigoplus_{j \in J(A)} \bigoplus_{k' \in K(B)} ((d_X \sim d_j) \wedge (d_Y \sim d_{k'})) \gg \gamma(a_j, a_{k'}) \odot B \oplus \\
&\qquad \bigoplus_{k \in K(A)} \bigoplus_{k' \in K(B)} ((d_X \sim d_k) \wedge (d_Y \sim d_{k'})) \gg \gamma(a_k, a_{k'}) \odot Z_4 \oplus \\
&\qquad \bigoplus_{l \in L(A)} \bigoplus_{l' \in L(B)} \left(\left(d_X \sim d_l \sim c_{l_{jmp}}\right) \wedge \left(d_Y \sim d_{l'} \sim c_{l'_{jmp}}\right) \right) \gg (c_l \wedge c_{l'}) \rhd Z_{7,l,l'}
\end{aligned}
$$

$$
Z_2 \quad \approx \quad A \parallel (d_Y \gg B), \text{ which is similar to } Z_1
$$

$$
Z_3 \quad \approx \quad (d_X \gg A) \parallel B, \text{ which is similar to } Z_1
$$

$$
Z_4 \quad \approx \quad A \parallel B, \text{ which is similar to } Z_1
$$

$$
\begin{aligned}
Z_5 \quad &\approx \quad d_X \gg A \\
&\approx \quad \bigoplus_{j \in J(A)} (d_X \sim d_j) \gg a_j \oplus \\
&\qquad \bigoplus_{k \in K(A)} (d_X \sim d_k) \gg a_k \odot A \oplus \\
&\qquad \bigoplus_{l \in L(A)} (d_X \sim d_l) \gg c_l \rhd A
\end{aligned}
$$

$$
Z_6 \quad \approx \quad d_Y \gg B, \text{ which is similar to } Z_5
$$

$$
Z_{7,l,l'} \quad \approx \quad A \between (c_{l'} \blacktriangleright B) \oplus B \between (c_l \blacktriangleright A) \oplus A \mid (c_{l'} \blacktriangleright B) \oplus B \mid (c_l \blacktriangleright A)
$$

$$
\begin{aligned}
\approx\quad & \bigoplus_{j\in J(A)} d_j \gg a_j \odot Z_{8,l'} \oplus \\
& \bigoplus_{k\in K(A)} d_k \gg a_k \odot Z_{9,l'} \oplus \\
& \bigoplus_{j'\in J(B)} d_{j'} \gg a_{j'} \odot Z_{10,l} \oplus \\
& \bigoplus_{k'\in K(B)} d_{k'} \gg a_{k'} \odot Z_{11,l} \oplus \\
& \bigoplus_{l''\in L(A)} ((d_{l''} \sim c_{l''_{jmp}}) \wedge c_{l'_{jmp}}) \gg (c_{l''} \wedge c_{l'}) \rhd Z_{7,l'',l'} \oplus \\
& \bigoplus_{l''\in L(B)} ((d_{l''} \sim c_{l''_{jmp}}) \wedge c_{l_{jmp}}) \gg (c_{l''} \wedge c_{l}) \rhd Z_{7,l,l''} \oplus \\
& \bigoplus_{j\in J(A)} \bigoplus_{j'\in J(B)} (d_j \wedge d_{j'}) \gg \gamma(a_j, a_{j'}) \oplus \\
& \bigoplus_{k\in K(A)} \bigoplus_{j'\in J(B)} (d_k \wedge d_{j'}) \gg \gamma(a_k, a_{j'}) \odot A \oplus \\
& \bigoplus_{j\in J(A)} \bigoplus_{k'\in K(B)} (d_j \wedge d_{k'}) \gg \gamma(a_j, a_{k'}) \odot B \oplus \\
& \bigoplus_{k\in K(A)} \bigoplus_{k'\in K(B)} (d_k \wedge d_{k'}) \gg \gamma(a_k, a_{k'}) \odot Z_4 \oplus \\
& \bigoplus_{l''\in L(A)} \bigoplus_{l'''\in L(B)} ((d_{l''} \sim c_{l''_{jmp}}) \wedge (d_{l'''} \sim c_{l'''_{jmp}})) \gg (c_{l''} \wedge c_{l'''}) \rhd Z_{7,l'',l'''}
\end{aligned}
$$

$$
\begin{aligned}
Z_{8,l'} \quad \approx\quad & c_{l'} \blacktriangleright B \\
\approx\quad & [true] \gg c_{l'} \rhd B \oplus rhs(B)
\end{aligned}
$$

$$
\begin{aligned}
Z_{9,l'} \quad \approx\quad & A \parallel (c_{l'} \blacktriangleright B) \\
\approx\quad & \bigoplus_{j\in J(A)} d_j \gg a_j \odot Z_{8,l'} \oplus \\
& \bigoplus_{k\in K(A)} d_k \gg a_k \odot Z_{9,l'} \oplus \\
& \bigoplus_{j'\in J(B)} d_{j'} \gg a_{j'} \odot A \oplus \\
& \bigoplus_{k'\in K(B)} d_{k'} \gg a_{k'} \odot Z_4 \oplus \\
& \bigoplus_{j\in J(A)} \bigoplus_{j'\in J(B)} (d_j \wedge d_{j'}) \gg \gamma(a_j, a_{j'}) \oplus \\
& \bigoplus_{k\in K(A)} \bigoplus_{j'\in J(B)} (d_k \wedge d_{j'}) \gg \gamma(a_k, a_{j'}) \odot A \oplus \\
& \bigoplus_{j\in J(A)} \bigoplus_{k'\in K(B)} (d_j \wedge d_{k'}) \gg \gamma(a_j, a_{k'}) \odot B \oplus \\
& \bigoplus_{k\in K(A)} \bigoplus_{k'\in K(B)} (d_k \wedge d_{k'}) \gg \gamma(a_k, a_{k'}) \odot Z_4 \oplus
\end{aligned}
$$

$$\bigoplus_{l'' \in L(A)} \bigoplus_{l''' \in L(B)} ((d_{l''} \sim c_{l''_{jmp}}) \wedge (d_{l'''} \sim c_{l'''_{jmp}})) \gg (c_{l''} \wedge c_{l'''}) \rhd Z_{7,l'',l'''} \oplus$$
$$\bigoplus_{l'' \in L(A)} ((d_{l''} \sim c_{l''_{jmp}}) \wedge c_{l'_{jmp}}) \gg (c_{l''} \wedge c_{l'}) \rhd Z_{7,l'',l'}$$

$Z_{10,l} \approx c_l \blacktriangleright A$, which is similar to $Z_{8,l'}$

$Z_{11,l} \approx B \parallel (c_l \blacktriangleright A)$, which is similar to $Z_{9,l'}$

3. Now, there are several linear recursive equations, namely A, B and Z_1 to $Z_{11,l}$. Since a linear recursive equation is also a semi-linear recursive equation, these linear equations can be transformed to abstracted-linear equations using the transformation of the previous stage. The result of this transformation is that A, B and Z_1 to $Z_{11,l}$ are now abstracted-linear. They are of the form $[\![v \mid d \gg C]\!]$, where v is a fresh variable and C is a linear recursive equation.
$$E = \left\{ \begin{array}{l} A \approx abslin(C), B \approx abslin(C), \\ X \approx [\![S_X \mid d_X \gg A]\!], Y \approx [\![S_Y \mid d_Y \gg B]\!], \\ P_0 \approx [\![S_{P_0} \mid d_{P_0} \gg A]\!], \ldots, P_n \approx [\![S_{P_n} \mid d_{P_n} \gg A]\!] \end{array} \right\}$$
and $C \approx lin(C), Z \approx [\![S_Z \mid d_Z \gg Z_1]\!], Z_1 \approx abslin(C), \ldots, Z_{11,l} \approx abslin(C)$.
4. The recursive equations in E are not abstracted-linear anymore, because A and B are now abstracted-linear. Moreover, Z is not abstracted-linear anymore, because Z_1 is now abstracted-linear as well. This is solved by applying the following transformation once to all recursive equations in E (except A and B) and to Z. Suppose $X \approx [\![S_X \mid d_X \gg A]\!]$ and $A \approx [\![V \mid d \gg C]\!]$:
$$\begin{array}{rcl} X & \approx & [\![S_X \mid d_X \gg A]\!] \\ & \approx & [\![S_X \mid d_X \gg [\![V \mid d \gg C]\!]]\!] \\ & \approx & [\![S_X \mid [\![V \mid d_X \gg d \gg C]\!]]\!] \\ & \approx & [\![S_X \cup V \mid (d_X \sim d) \gg C]\!] \end{array}$$
Now
$$E = \left\{ \begin{array}{l} A \approx abslin(C), B \approx abslin(C), X \approx abslin(C), Y \approx abslin(C), \\ P_0 \approx abslin(C), \ldots, P_n \approx abslin(C) \end{array} \right\}$$
and $C \approx lin(C), Z \approx abslin(C), Z_1 \approx abslin(C), \ldots, Z_{11,l} \approx abslin(C)$.
5. Finally, Z and C are added to E, and A and B are deleted from E because they are not used anymore. Now, all recursive equations in E are abstracted-linear, except for the single equation C which is linear. Now
$$E = \left\{ \begin{array}{l} C \approx lin(C), X \approx abslin(C), Y \approx abslin(C), Z \approx abslin(C), \\ P_0 \approx abslin(C), \ldots, P_n \approx abslin(C) \end{array} \right\}$$

The result of the *elim_par_comp* function is the single recursion variable Z.

Lemma 26 (Resulting form) *The result of transformation elim_par_comp is a single abstracted-linear recursion variable. Furthermore, after transformation, all recursive equations in E are abstracted-linear except for the single equation C which is linear.*

Proof The result of *elim_par_comp* is the single recursion variable Z. Z is introduced in step 2 above and it is straightforward to verify that Z is abstracted-linear at that point. In step 3, Z becomes the abstraction of the abstracted-linear variable Z_1. Then in step 4 this abstraction

is merged with the abstraction in Z_1, which makes Z abstracted-linear again. Finally, step 5 does not affect Z, so the resulting recursion variable Z is abstracted-linear.

By following the steps 1 to 5 above it is straightforward to see that, after transformation, indeed all recursive equations in E are abstracted-linear except for the single equation C which is linear. ⊠

Lemma 27 (Soundness) *The transformation elim_par_comp is sound and elim_par_comp*$(X, Y) \approx X \parallel Y$ *for abstracted-linear recursion variables* X *and* Y*.*

Proof We prove soundness of *elim_par_comp* by proving that each step is sound:

1. Only axioms of abstraction are used.
2. It is tedious but straightforward to verify that Z is indeed equal to the parallel composition of X and Y.
3. Soundness of this transformation was proven to be sound in the previous section.
4. Only axioms of abstraction are used.
5. As can be seen in the contents of E after step 4, A and B are not used anymore, so they can be deleted without affecting any other equations.

We conclude that *elim_par_comp* is sound.

The result of the *elim_par_comp* function is the single recursion variable Z. In step 2, Z is defined such that it is robustly bisimilar to $X \parallel Y$. Every subsequent step either does not affect Z at all or it transforms Z into an equivalent process. Therefore, *elim_par_comp*$(X, Y) = Z \approx X \parallel Y$. ⊠

Lemma 28 (Well-definedness) *elim_par_comp is well-defined.*

Proof The result of the computation of Z in step 2 is finite. The transformation from linear equations to abstracted-linear equations in step 3 was proven to be well-defined in the previous section. Well-definedness of the other steps is trivial. Therefore, *elim_par_comp* is well-defined. ⊠

Eliminating Encapsulation The *elim_encap* transformation takes a set of discrete actions H and an abstracted-linear recursion variable X as its parameters. The result is a single abstracted-linear recursion variable Z that is equivalent to $\partial_H(X)$.

Suppose X is abstracted-linear, and the encapsulation in $\partial_H(X)$ needs to be eliminated. Then, X has the following form, where A is a linear recursive equation:

$$\begin{array}{lcl} X & \approx & [\![\, S \mid d_X \gg A \,]\!] \\ A & \approx & \bigoplus_{j \in J(A)} d_j \gg a_j \oplus \bigoplus_{k \in K(A)} d_k \gg a_k \odot A \oplus \bigoplus_{l \in L(A)} d_l \gg c_l \rhd A \end{array}$$

As in the elimination of parallel composition, all equations in the set of equations E of the specification at this point are abstracted-linear, except for the linear equation for A. Therefore, before eliminating encapsulation:

$$E = \{\ A \approx \mathit{lin}(A), X \approx \mathit{abslin}(A), P_0 \approx \mathit{abslin}(A), \ldots, P_n \approx \mathit{abslin}(A)\ \}$$

The following steps are performed in the order they are listed:

1. Two new recursive equations Z and B are introduced and added to E:

$$E := E \cup \{Z \approx [\![S \mid d_X \gg B]\!], B \approx \mathit{elim\_encap}_2(H, \mathit{rhs}(A))\}$$

where the $\mathit{elim\_encap}_2$ function is defined as follows:

$$\begin{array}{lcl}
\mathit{elim_encap}_2(H, p \oplus q) & = & \mathit{elim_encap}_2(p) \oplus \mathit{elim_encap}_2(q) \\
\mathit{elim_encap}_2(H, d \gg a) & = & \begin{cases} \delta & \text{if } a \in H \\ d \gg a & \text{if } a \notin H \end{cases} \\
\mathit{elim_encap}_2(H, d \gg a \odot A) & = & \begin{cases} \delta & \text{if } a \in H \\ d \gg a \odot B & \text{if } a \notin H \end{cases} \\
\mathit{elim_encap}_2(H, d \gg c \rhd A) & = & d \gg c \rhd B \\
\mathit{elim_encap}_2(H, \delta) & = & \delta
\end{array}$$

Then

$$E = \left\{ \begin{array}{l} Z \approx \mathit{abslin}(B), A \approx \mathit{lin}(A), B \approx \mathit{lin}(B), X \approx \mathit{abslin}(A), \\ P_0 \approx \mathit{abslin}(A), \ldots, P_n \approx \mathit{abslin}(A) \end{array} \right\}$$

2. Now, there are two linear recursive equations in E, namely A and B. As in the *elim_par_comp* transformation, the transformation of the previous stage is applied to these two equations to make them abstracted-linear. They become of the form $[\![v \mid d \gg C]\!]$, where v is a fresh variable and C is a linear equation. Now

$$E = \left\{ \begin{array}{l} Z \approx [\![S_Z \mid d_Z \gg B]\!], A \approx \mathit{abslin}(C), \\ B \approx \mathit{abslin}(C), X \approx [\![S_X \mid d_X \gg A]\!], \\ P_0 \approx [\![S_{P_0} \mid d_{P_0} \gg A]\!], \ldots, P_n \approx [\![S_{P_n} \mid d_{P_n} \gg A]\!] \end{array} \right\}$$

and $C \approx \mathit{lin}(C)$.

3. As in the *elim_par_comp* transformation, the recursive equations in E are not abstracted-linear anymore, because A and B are now abstracted-linear. The same transformation as in the *elim_par_comp* transformation is applied to all recursive equations in E, except the equations of A and B, to make them abstracted-linear again:

$$\begin{array}{rcl}
X & \approx & [\![S \mid d_X \gg A]\!] \\
& \approx & [\![S \mid d_X \gg [\![v \mid d \gg C]\!]]\!] \\
& \approx & [\![S \mid [\![v \mid d_X \gg d \gg C]\!]]\!] \\
& \approx & [\![S \cup \{v\} \mid (d_X \sim d) \gg C]\!]
\end{array}$$

Now

$$E = \left\{ \begin{array}{l} Z \approx \mathit{abslin}(C), A \approx \mathit{abslin}(C), B \approx \mathit{abslin}(C), X \approx \mathit{abslin}(C), \\ P_0 \approx \mathit{abslin}(C), \ldots, P_n \approx \mathit{abslin}(C) \end{array} \right\}$$

and $C \approx \mathit{lin}(C)$.

4. Finally, C is added to E, and A and B are deleted from E. All recursive equations in E are now abstracted-linear, except for the single equation C which is linear. Now

$$E = \left\{ C \approx \mathit{lin}(C), Z \approx \mathit{abslin}(C), X \approx \mathit{abslin}(C), P_0 \approx \mathit{abslin}(C), \ldots, P_n \approx \mathit{abslin}(C) \right\}$$

The result of the *elim_encap* function is the single recursion variable Z.

Lemma 29 $B \approx \partial_H(A)$.

Proof We prove this using RSP, by showing that $\partial_H(A)$ and B are both solutions of another guarded recursive equation.

$\partial_H(A) \approx \partial_H(\mathit{rhs}(A))$

$$\begin{aligned}
&\approx \partial_H\left(\bigoplus_{j\in J(A)} d_j \gg a_j \oplus \bigoplus_{k\in K(A)} d_k \gg a_k \odot A \oplus \bigoplus_{l\in L(A)} d_l \gg c_l \rhd A\right)\\
&\approx \bigoplus_{j\in J(A)} \partial_H(d_j \gg a_j) \oplus \bigoplus_{k\in K(A)} \partial_H(d_k \gg a_k \odot A) \oplus \bigoplus_{l\in L(A)} \partial_H(d_l \gg c_l \rhd A)\\
&\approx \bigoplus_{j\in J(A)\wedge a_j\notin H} d_j \gg a_j \oplus \bigoplus_{k\in K(A)\wedge a_k\notin H} d_k \gg a_k \odot \partial_H(A) \oplus \bigoplus_{l\in L(A)} d_l \gg c_l \rhd \partial_H(A)
\end{aligned}$$

$$\begin{aligned}
B &\approx elim_encap_2(H, rhs(A))\\
&\approx elim_encap_2\left(H, \begin{array}{ll}\bigoplus_{j\in J(A)} & d_j \gg a_j \oplus\\ \bigoplus_{k\in K(A)} & d_k \gg a_k \odot A \oplus\\ \bigoplus_{l\in L(A)} & d_l \gg c_l \rhd A\end{array}\right)\\
&\approx \bigoplus_{j\in J(A)} elim_encap_2(H, d_j \gg a_j) \oplus\\
&\quad \bigoplus_{k\in K(A)} elim_encap_2(H, d_k \gg a_k \odot A) \oplus\\
&\quad \bigoplus_{l\in L(A)} elim_encap_2(H, d_l \gg c_l \rhd A)\\
&\approx \bigoplus_{j\in J(A)\wedge a_j\notin H} d_j \gg a_j \oplus \bigoplus_{k\in K(A)\wedge a_k\notin H} d_k \gg a_k \odot B \oplus \bigoplus_{l\in L(A)} d_l \gg c_l \rhd B
\end{aligned}$$

We define the equation Y as follows:

$$Y \approx \bigoplus_{j\in J(A)\wedge a_j\notin H} d_j \gg a_j \oplus \bigoplus_{k\in K(A)\wedge a_k\notin H} d_k \gg a_k \odot Y \oplus \bigoplus_{l\in L(A)} d_l \gg c_l \rhd Y$$

We see that both $\partial_H(A)$ and B are a solution of the guarded equation Y, so by RSP we conclude that $B \approx \partial_H(A)$. ⊠

Lemma 30 (Soundness) *The transformation elim_encap is sound, that is elim_encap$(H, X) \approx \partial_H(X)$ for any abstracted-linear recursion variable X.*

Proof We prove soundness of *elim_encap* by proving that each step is sound:

1. We prove that $Z \approx \partial_H(X)$, using lemma 29:

$$\begin{aligned}
Z &\approx [\![S \mid d_X \gg B]\!]\\
&\approx [\![S \mid d_X \gg \partial_H(A)]\!]\\
&\approx [\![S \mid \partial_H(d_X \gg A)]\!]\\
&\approx \partial_H([\![S \mid d_X \gg A]\!])\\
&\approx \partial_H(X)
\end{aligned}$$

   elim_encap$(H, X) = Z$ and $Z \approx \partial_H(X)$, so *elim_encap$(H, X) \approx \partial_H(X)$* for any abstracted-linear recursion variable X.
2. Soundness of this transformation was proven in the previous section.
3. Only axioms of abstraction are used.
4. As can be seen in the contents of E after step 3, A and B are not used anymore, so they can be deleted without affecting any other equations.

We conclude that *elim_encap* is sound. ⊠

Lemma 31 (Resulting form) *The result of the transformation elim_encap is a single abstracted-linear recursion variable. Furthermore, after transformation, all recursive equations in E are abstracted-linear except for the single equation C which is linear.*

Proof The result of *elim_par_comp* is the single recursion variable Z. In the proof of lemma 29 we see that

$$B \approx \bigoplus_{j \in J(A) \wedge a_j \notin H} d_j \gg a_j \oplus \bigoplus_{k \in K(A) \wedge a_k \notin H} d_k \gg a_k \odot B \oplus \bigoplus_{l \in L(A)} d_l \gg c_l \rhd B$$

Clearly, B is linear after step 1. Since $Z \approx [\![S \mid d_X \gg B]\!]$, we conclude that Z is abstracted-linear after step 1. In step 2, Z becomes the abstraction of the abstracted-linear variable B. Then in step 3 this abstraction is merged with the abstraction in B, which makes Z abstracted-linear again. Finally, step 4 does not affect Z, so the resulting recursion variable Z is abstracted-linear.

By following the steps 1 to 4 above it is straightforward to see that, after transformation, indeed all recursive equations in E are abstracted-linear except for the single equation C which is linear. ⊠

Lemma 32 (Well-definedness) *elim_encap is well-defined.*

Proof The transformation from linear equations to abstracted-linear equations was proven to be well-defined in the previous section. Well-definedness of the other steps is trivial. Therefore, *elim_encap* is well-defined. ⊠

First, the proofs for the *elim_par_comp* and *elim_encap* functions have been presented. Now, the proofs for the function T_1 are presented, because lemmas about the *elim_par_comp* and *elim_encap* functions are used in these proofs.

Lemma 33 (Soundness) *T_1 and the final transformation are sound.*

Proof First, we prove that $T_2(p) \approx p$ for any p is of the form $HyPA_{\mathrm{par}}$, by induction on the structure of the form $HyPA_{\mathrm{par}}$:

- X: trivial.
- $p \parallel q$, with p and q of the form $HyPA_{\mathrm{par}}$: Applying the induction hypothesis twice gives $T_2(p) \approx p$ and $T_2(q) \approx q$. Furthermore, it is straightforward to see that $T_2(r)$ always returns a single abstracted-linear recursion variable X for any term r of the form $HyPA_{\mathrm{par}}$. Therefore, using lemma 27, $T_2(p \parallel q) = \mathit{elim\_par\_comp}(p, q) \approx p \parallel q$.
- $\partial_H(p)$: Applying the induction hypothesis gives $T_2(p) \approx p$. Furthermore, it is straightforward to see that $T_2(r)$ always returns a single abstracted-linear recursion variable X for any term r of the form $HyPA_{\mathrm{par}}$. Therefore, using lemma 30, $T_2(\partial_H(p)) = \mathit{elim\_encap}(H, p) \approx \partial_H(p)$.

Now, we prove that $T_1(p) \approx p$ for any p is of the form $HyPA_{\mathrm{par}}$:

- p is of the form $[\![\ V\ \mid q\ ]\!]$, where q is of the form $HyPA_{\mathrm{par}}$:

$$T_1(p) = T_1([\![\ V\ \mid q\ ]\!]) = [\![\ V\ \mid T_2(q)\ ]\!] \approx [\![\ V\ \mid q\ ]\!] \approx p.$$

- p is of not the form $[\![\ V\ \mid q\ ]\!]$: $T_1(p) = [\![\ \emptyset\ \mid T_2(p)\ ]\!] \approx [\![\ \emptyset\ \mid p\ ]\!] \approx p$.

Therefore, T_1 is sound. It is trivial to see that the final transformation $t = [\![\ V\ \mid X\ ]\!] \approx [\![\ V\ \mid [\![\ W\ \mid d \gg A\ ]\!]\ ]\!] \approx [\![\ V \cup W\ \mid d \gg A\ ]\!]$ is sound. ⊠

Lemma 34 (Well-definedness) *T_1 is well-defined.*

Proof It is straightforward to see that T_2 is well-defined, because recursion in the right-hand side only occurs on strictly smaller sub terms. Since T_2 is well-defined, it is trivial to see that T_1 is well-defined. ⊠

Lemma 35 (Resulting form) *After the transformations in this stage, the specification $\langle\, t \,|\, E \,\rangle$ is linear.*

Proof The result of $T_2(p)$ is always a single recursion variable, say X, whose right-hand side is abstracted-linear. Therefore, the result of $T_1(p)$ is always an abstraction of X, i.e. $[\![\ V\ \mid X\ ]\!]$. Then we have, $t = [\![\ V\ \mid X\ ]\!] \approx [\![\ V\ \mid [\![\ W\ \mid d \gg A\ ]\!]\ ]\!] \approx [\![\ V \cup W\ \mid d \gg A\ ]\!]$, where A is linear. Clearly, the resulting specification $\langle\, t \,|\, \{A\} \,\rangle$ is linear. ⊠

5 Optimization

The linearization algorithm was proven to be correct in the previous section. However, the number of summands in the resulting linear specification is enormous. An experiment showed that the linearization of the simple Thermostat example in section 2.4 leads to a linear specification that has 1678 summands. This number has to be reduced drastically to make our algorithm suitable for linearization of real-world models.

The experiment also showed that the problem lies in the step that combines multiple linear equations into a single linear equation (i.e. the transformation in stage 2). The problem is magnified at least quadratically when the parallel composition of two such linear equations is calculated. Therefore, the optimization efforts are focused on reducing the number and size of the summands in this linear equation.

5.1 Creating a Single Stack Clause per Summand

All summands in the linear specification that are the result of the transformation in stage 2 have the form $(s \sim d) \gg x$ where s denotes a re-initialization clause that only contains predicates on stack variables and d denotes a re-initialization clause that does *not* refer to stack variables. However, after eliminating parallel composition and merging all linear equations into a single new linear equation, the summands of the new linear equation do not have the form $(s \sim d) \gg x$ anymore. The goal of this step is to make these summands of the form $(s \sim d) \gg x$ again.

Conjecture 1 *For all terms x, all re-initialization clauses s and s' which only contain predicates on stack variables as used in the linearization algorithm and all re-initialization clauses d and d' which do not refer to stack variables, where $\equiv$ denotes equivalence of re-initialization clauses:*

$$\begin{aligned} (s \sim d) \wedge (s' \sim d') &\equiv (s \wedge s') \sim (d \wedge d') && (1)\\ (s \sim d) \wedge d' &\equiv s \sim (d \wedge d') && (2) \end{aligned}$$

Theorem 4 *After elimination of a parallel composition, the specification contains a single linear equation. All summands in this equation can be transformed to the form $(s \sim d) \gg x$ where s denotes a re-initialization clause that only contains predicates on stack variables and d denotes a re-initialization clause that does not refer to stack variables.*

Proof Lemma 26 says that, after elimination of a parallel composition, the specification contains a single linear equation. First we show which forms the summands in this linear equation may have and then we show that each of these forms can be transformed into the form $(s \sim d) \gg x$.

As explained in section 4.3.3, this linear equation is created by merging the processes A, B and Z_1 to $Z_{11,l}$. The re-initialization clauses in the processes A, B and Z_1 to $\mathrm{Z}_{11,\mathrm{l}}$ are of one of the following forms:

$$\begin{array}{ll} s \sim d \gg x & (s \sim s' \sim d) \wedge (s'' \sim s''' \sim d') \gg x \\ s \sim (s' \sim d) \gg x & (s \sim d \sim d') \wedge (s' \sim d'' \sim d''') \gg x \\ (s \sim d \sim d') \wedge d'' \gg x & (s \sim s' \sim d \sim d') \wedge (s'' \sim s''' \sim d'' \sim d''') \gg x \\ (s \sim d) \wedge (s' \sim d') \gg x & \end{array}$$

Then, when these equations are merged into one single linear equation, all summands are prefixed with a re-initialization clause that only contains a predicate on the newly introduced stack variable. Therefore, the summands in the new linear equation are of one of the following forms:

1. $s' \sim (s \sim d) \gg x$
2. $s'' \sim (s \sim (s' \sim d)) \gg x$
3. $s' \sim ((s \sim d \sim d') \wedge d'') \gg x$
4. $s'' \sim ((s \sim d) \wedge (s' \sim d')) \gg x$
5. $s'''' \sim ((s \sim s' \sim d) \wedge (s'' \sim s''' \sim d')) \gg x$
6. $s'' \sim ((s \sim d \sim d') \wedge (s' \sim d'' \sim d''')) \gg x$
7. $s'''' \sim ((s \sim s' \sim d \sim d') \wedge (s'' \sim s''' \sim d'' \sim d''')) \gg x$

Now, we show that each of these forms can be transformed into the form $(s \sim d) \gg x$, using only Conjecture 1, associativity of concatenation and conjunction, and commutativity of conjunction:

1. $s' \sim (s \sim d) \gg x \approx (s' \sim s) \sim d \gg x$
2. $\begin{aligned} & s'''' \sim ((s \sim s' \sim d) \wedge (s'' \sim s''' \sim d')) \gg x \\ \approx\ & s'''' \sim (((s \sim s') \sim d) \wedge ((s'' \sim s''') \sim d')) \gg x \\ \approx\ & s'''' \sim (((s \sim s') \wedge (s'' \sim s''')) \sim (d \wedge d')) \gg x \\ \approx\ & (s'''' \sim ((s \sim s') \wedge (s'' \sim s'''))) \sim (d \wedge d') \gg x \end{aligned}$
3. $s'' \sim (s \sim (s' \sim d)) \gg x \approx (s'' \sim s \sim s') \sim d \gg x$
4. $\begin{aligned} & s'' \sim ((s \sim d \sim d') \wedge (s' \sim d'' \sim d''')) \gg x \\ \approx\ & s'' \sim ((s \sim (d \sim d')) \wedge (s' \sim (d'' \sim d'''))) \gg x \\ \approx\ & s'' \sim ((s \wedge s') \sim ((d \sim d') \wedge (d'' \sim d'''))) \gg x \\ \approx\ & (s'' \sim (s \wedge s')) \sim ((d \sim d') \wedge (d'' \sim d''')) \gg x \end{aligned}$
5. $\begin{aligned} & s' \sim ((s \sim d \sim d') \wedge d'') \gg x \\ \approx\ & s' \sim ((s \sim (d \sim d')) \wedge d'') \gg x \\ \approx\ & s' \sim (s \sim ((d \sim d') \wedge d'')) \gg x \\ \approx\ & (s' \sim s) \sim ((d \sim d') \wedge d'') \gg x \end{aligned}$

$$
\begin{array}{lll}
6. & & s'''' \sim ((s \sim s' \sim d \sim d') \wedge (s'' \sim s''' \sim d'' \sim d''')) \gg x \\
& \approx & s'''' \sim (((s \sim s') \sim (d \sim d')) \wedge ((s'' \sim s''') \sim (d'' \sim d''''))) \gg x \\
& \approx & s'''' \sim (((s \sim s') \wedge (s'' \sim s''')) \sim ((d \sim d') \wedge (d'' \sim d''''))) \gg x \\
& \approx & (s'''' \sim ((s \sim s') \wedge (s'' \sim s''''))) \sim ((d \sim d') \wedge (d'' \sim d'''')) \gg x \\
7. & & s'' \sim ((s \sim d) \wedge (s' \sim d')) \gg x \\
& \approx & s'' \sim ((s \wedge s') \sim (d \wedge d')) \gg x \\
& \approx & (s'' \sim (s \wedge s')) \sim (d \wedge d') \gg x
\end{array}
$$

$\boxtimes$

5.2 Merging Summands

Creating a single stack clause per summand did not make the size of the single linear equation smaller, but that step prepared the linear equation for this step, which merges summands. In this step, the rewrite system consisting of the following rule is applied to the right-hand side of the linear equation:

$$s \sim d \gg x \oplus s' \sim d \gg x \rightarrow (s \vee s') \sim d \gg x$$

Soundness of this rule is straightforward to prove using the axioms $d \gg x \oplus d' \gg x \approx (d \vee d') \gg x$ and $d \sim d' \gg x \approx d \gg d' \gg x$ and termination is trivial.

Experiments show that this step drastically reduces the number of summands in the linear equation. Using this optimization, the number of summands in the resulting linear specification of the Thermostat example was reduced from 1678 to only 5 (see the next section for the linearization of this example). However, after this optimization, the re-initialization clauses on stack variables are very large. We strongly feel that these clauses can be optimized further, by trying to simplify their contents. This optimization is not a core part of the algorithm though, so this is considered to be future work.

5.3 Eliminating Superfluous Clauses

A final simple step is to eliminate all superfluous $[true]$, $[false]$ and c_{jmp} re-initialization clauses. This is achieved by applying the rewrite system consisting of the following rules to the right-hand side of the linear equation:

$$
\begin{array}{lcl@{\qquad}lcl}
x \oplus \delta & \rightarrow & x & [false] \vee d & \rightarrow & d \\
\delta \oplus x & \rightarrow & x & [true] \vee d & \rightarrow & [true] \\
[false] \gg x & \rightarrow & \delta & d \vee [false] & \rightarrow & d \\
[true] \gg x & \rightarrow & x & d \vee [true] & \rightarrow & [true] \\
[false] \wedge d & \rightarrow & [false] & [false] \sim d & \rightarrow & [false] \\
[true] \wedge d & \rightarrow & d & [true] \sim d & \rightarrow & d \\
d \wedge [false] & \rightarrow & [false] & d \sim [false] & \rightarrow & [false] \\
d \wedge [true] & \rightarrow & d & d \sim [true] & \rightarrow & d \\
(c_{jmp} \wedge c'_{jmp}) \gg (c \wedge c') & \rightarrow & c \wedge c' & & &
\end{array}
$$

It is straightforward to verify that these rewrite rules are sound, except for the rule for c_{jmp} clauses. Soundness of this rule can be proven by calculation on re-initialization clauses and the axiom $c_{jmp} \gg c \approx c$. Furthermore, the rewrite system is terminating, because the right-hand side of every rule is strictly smaller than its left-hand side.

6 Example

In this section, the example in section 2.4 is linearized to illustrate the linearization algorithm. The model of this example is the following:

$$
\begin{array}{lcl}
\text{HeaterOn} & \approx & (x|\,\dot{x} = -x + 4) \blacktriangleright \mathit{rOff} \odot \text{HeaterOff} \\
\text{HeaterOff} & \approx & (x|\,\dot{x} = -x) \blacktriangleright \mathit{rOn} \odot \text{HeaterOn} \\
\text{Thermostat} & \approx & (x_{min} \leq x \leq x_{max}) \blacktriangleright \left(\begin{array}{l} [x = x_{min}] \gg \mathit{sOn} \odot \text{Thermostat} \\ \oplus \\ [x = x_{max}] \gg \mathit{sOff} \odot \text{Thermostat} \end{array} \right) \\
\text{Controller} & \approx & \partial_H\,(\text{HeaterOn} \parallel \text{Thermostat}) \\
 & & \\
\gamma(\mathit{sOn}, \mathit{rOn}) & = & \mathit{On} \\
\gamma(\mathit{sOff}, \mathit{rOff}) & = & \mathit{Off} \\
H & = & \{\mathit{sOff}, \mathit{sOn}, \mathit{rOff}, \mathit{rOn}\}
\end{array}
$$

The HyPA_{lin} specification of this model is therefore

$$\langle\, \partial_H\,(\text{HeaterOn} \parallel \text{Thermostat}) \mid E \,\rangle$$

where E consists of the previously given equations.

6.1 Stage 1: Transforming Equations into Semi-linear Form

The right-hand sides of all equations in the specification are transformed to semi-linear form. This results in the following specification:

$$\langle\, \partial_H\,(\text{HeaterOn} \parallel \text{Thermostat}) \mid E_1 \,\rangle$$

where E_1 consists of the following equations

$$
\begin{array}{lcl}
\text{Thermostat} & \approx & [\mathit{true}] \gg (x_{min} \leq x \leq x_{max}) \rhd \text{X2} \\
 & \oplus & [x = x_{min}] \gg \mathit{sOn} \odot \text{Thermostat} \\
 & \oplus & [x = x_{max}] \gg \mathit{sOff} \odot \text{Thermostat} \\
\text{X2} & \approx & [x = x_{min}] \gg \mathit{sOn} \odot \text{Thermostat} \oplus [x = x_{max}] \gg \mathit{sOff} \odot \text{Thermostat} \\
\text{HeaterOff} & \approx & [\mathit{true}] \gg (x|\,\dot{x} = -x) \rhd \text{X4} \oplus [\mathit{true}] \gg \mathit{rOn} \odot \text{HeaterOn} \\
\text{X4} & \approx & [\mathit{true}] \gg \mathit{rOn} \odot \text{HeaterOn} \\
\text{HeaterOn} & \approx & [\mathit{true}] \gg (x|\,\dot{x} = -x + 4) \rhd \text{X6} \oplus [\mathit{true}] \gg \mathit{rOff} \odot \text{HeaterOff} \\
\text{X6} & \approx & [\mathit{true}] \gg \mathit{rOff} \odot \text{HeaterOff}
\end{array}
$$

6.2 Stage 2: From Semi-linear to Abstracted-linear

Now, the equations of the previous stage are combined into one linear equation A_0 and all equations in the specification are transformed into abstracted-linear form. Note that A_0 is optimized using the optimizations of the previous section. This stage results in the following specification:

$$\langle\, \partial_H\,(\text{HeaterOn} \parallel \text{Thermostat}) \mid E_2 \,\rangle$$

where E_2 consists of the following equations

$$
\begin{array}{lcl}
\text{HeaterOff} & \approx & \left\| \left[s_0 \;\middle|\; \left[s_0 \;\middle|\; s_0^+ = \mathit{push}(\text{HeaterOff}, \emptyset) \right] \gg \text{A}_0 \right] \right\| \\
\text{X4} & \approx & \left\| \left[s_0 \;\middle|\; \left[s_0 \;\middle|\; s_0^+ = \mathit{push}(\text{X4}, \emptyset) \right] \gg \text{A}_0 \right] \right\| \\
\text{HeaterOn} & \approx & \left\| \left[s_0 \;\middle|\; \left[s_0 \;\middle|\; s_0^+ = \mathit{push}(\text{HeaterOn}, \emptyset) \right] \gg \text{A}_0 \right] \right\| \\
\text{X6} & \approx & \left\| \left[s_0 \;\middle|\; \left[s_0 \;\middle|\; s_0^+ = \mathit{push}(\text{X6}, \emptyset) \right] \gg \text{A}_0 \right] \right\| \\
\text{Thermostat} & \approx & \left\| \left[s_0 \;\middle|\; \left[s_0 \;\middle|\; s_0^+ = \mathit{push}(\text{Thermostat}, \emptyset) \right] \gg \text{A}_0 \right] \right\| \\
\text{X2} & \approx & \left\| \left[s_0 \;\middle|\; \left[s_0 \;\middle|\; s_0^+ = \mathit{push}(\text{X2}, \emptyset) \right] \gg \text{A}_0 \right] \right\|
\end{array}
$$

and

$$
\begin{aligned}
&\mathrm{A}_0 \\
\approx\ & \left(\begin{array}{l} \left[\begin{array}{c|l} s_0 & s_0 \neq \emptyset \wedge get(s_0{}^-) = \mathrm{HeaterOff} \\ & \wedge\ s_0{}^+ = push(\mathrm{HeaterOn}, pop(s_0{}^-)) \end{array} \right] \vee \\ [s_0 \,|s_0 \neq \emptyset \wedge get(s_0{}^-) = \mathrm{X4} \wedge s_0{}^+ = push(\mathrm{HeaterOn}, pop(s_0{}^-))] \end{array} \right) \gg rOn \odot \mathrm{A}_0 \\
\oplus\ & \left(\begin{array}{l} \left[\begin{array}{c|l} s_0 & s_0 \neq \emptyset \wedge get(s_0{}^-) = \mathrm{HeaterOn} \\ & \wedge\ s_0{}^+ = push(\mathrm{HeaterOff}, pop(s_0{}^-)) \end{array} \right] \vee \\ [s_0 \,|s_0 \neq \emptyset \wedge get(s_0{}^-) = \mathrm{X6} \wedge s_0{}^+ = push(\mathrm{HeaterOff}, pop(s_0{}^-))] \end{array} \right) \gg rOff \odot \mathrm{A}_0 \\
\oplus\ & \left(\begin{array}{l} \left[\begin{array}{c|l} s_0 & s_0 \neq \emptyset \wedge get(s_0{}^-) = \mathrm{Thermostat} \\ & \wedge\ s_0{}^+ = push(\mathrm{Thermostat}, pop(s_0{}^-)) \end{array} \right] \vee \\ [s_0 \,|s_0 \neq \emptyset \wedge get(s_0{}^-) = \mathrm{X2} \wedge s_0{}^+ = push(\mathrm{Thermostat}, pop(s_0{}^-))] \\ \sim [x = x_{min}] \end{array} \right) \gg sOn \odot \mathrm{A}_0 \\
\oplus\ & \left(\begin{array}{l} \left[\begin{array}{c|l} s_0 & s_0 \neq \emptyset \wedge get(s_0{}^-) = \mathrm{Thermostat} \\ & \wedge\ s_0{}^+ = push(\mathrm{Thermostat}, pop(s_0{}^-)) \end{array} \right] \vee \\ [s_0 \,|s_0 \neq \emptyset \wedge get(s_0{}^-) = \mathrm{X2} \wedge s_0{}^+ = push(\mathrm{Thermostat}, pop(s_0{}^-))] \\ \sim [x = x_{max}] \end{array} \right) \gg sOff \odot \mathrm{A}_0 \\
\oplus\ & \left[\begin{array}{c|l} s_0 & s_0 \neq \emptyset \wedge get(s_0{}^-) = \mathrm{HeaterOff} \\ & \wedge\ s_0{}^+ = push(\mathrm{X4}, pop(s_0{}^-)) \end{array} \right] \gg ((x|\, \dot{x} = -x) \wedge (\, s_0|\, \dot{s_0} = 0)) \rhd \mathrm{A}_0 \\
\oplus\ & \left[\begin{array}{c|l} s_0 & s_0 \neq \emptyset \wedge get(s_0{}^-) = \mathrm{HeaterOn} \\ & \wedge\ s_0{}^+ = push(\mathrm{X6}, pop(s_0{}^-)) \end{array} \right] \gg ((x|\, \dot{x} = -x + 4) \wedge (\, s_0|\, \dot{s_0} = 0)) \rhd \mathrm{A}_0 \\
\oplus\ & \left[\begin{array}{c|l} s_0 & s_0 \neq \emptyset \wedge get(s_0{}^-) = \mathrm{Thermostat} \\ & \wedge\ s_0{}^+ = push(\mathrm{X2}, pop(s_0{}^-)) \end{array} \right] \gg ((x_{min} \leq x \leq x_{max}) \wedge (\, s_0|\, \dot{s_0} = 0)) \rhd \mathrm{A}_0
\end{aligned}
$$

6.3 Stage 3: Transforming the Initial Term

First, the parallel composition of HeaterOn and Thermostat is computed. Two new stack variables are introduced, namely one in the step where the stack variable of the Thermostat is renamed and one in the step where all equations are combined into one linear equation again. Note that a clause that contains only conditions on stack variables is denoted as a function $[sc_i(s_0, \ldots, s_n)]$, which represents a boolean predicate on the stack variables $s_0, \ldots, s_n$. The actual contents of these clauses are omitted, because these clauses are quite large at this point (one clause would fill a whole page). After this step, the optimized specification is the following:

$$\langle\, \partial_H\, (\mathrm{Z}_0) \mid \{\mathrm{Z}_0, \mathrm{A}_2\} \,\rangle$$

where

$$
\begin{aligned}
\mathrm{Z}_0 \quad \approx\ & \left| \left[\begin{array}{c|l} s_0, s_1, s_2 & \left(\begin{array}{l} [s_0 \,|s_0^+ = push(\mathrm{HeaterOn}, \emptyset)\,] \sim \\ [s_1 \,|s_1^+ = push(\mathrm{Thermostat}, \emptyset)\,] \sim \\ [s_2 \,|s_2^+ = push(\mathrm{Z}_0, \emptyset)\,] \end{array} \right) \gg \mathrm{A}_2 \end{array} \right] \right| \\
\mathrm{A}_2 \quad \approx\ & [sc_0(s_0, s_1, s_2)] \sim [x = x_{max}] \gg Off \odot \mathrm{A}_2 \\
\oplus\ & [sc_1(s_0, s_1, s_2)] \sim [x = x_{min}] \gg On \odot \mathrm{A}_2 \\
\oplus\ & [sc_2(s_0, s_1, s_2)] \sim [x = x_{min}] \gg sOn \odot \mathrm{A}_2 \\
\oplus\ & [sc_3(s_0, s_1, s_2)] \sim [x = x_{max}] \gg sOff \odot \mathrm{A}_2 \\
\oplus\ & [sc_4(s_0, s_1, s_2)] \gg rOff \odot \mathrm{A}_2 \\
\oplus\ & [sc_5(s_0, s_1, s_2)] \gg rOn \odot \mathrm{A}_2 \\
\oplus\ & [sc_6(s_0, s_1, s_2)] \gg \left(\begin{array}{l} (x|\, \dot{x} = -x + 4) \wedge (x_{min} \leq x \leq x_{max}) \\ \wedge\ (\, s_0|\, \dot{s_0} = 0) \wedge (\, s_1|\, \dot{s_1} = 0) \wedge (\, s_2|\, \dot{s_2} = 0) \end{array} \right) \rhd \mathrm{A}_2
\end{aligned}
$$

$$\oplus \quad [sc_7(s_0,s_1,s_2)] \gg \left(\begin{array}{l} (x|\,\dot{x} = -x) \wedge (x_{min} \leq x \leq x_{max}) \\ \wedge\,(s_0|\,\dot{s_0} = 0) \wedge (s_1|\,\dot{s_1} = 0) \wedge (s_2|\,\dot{s_2} = 0) \end{array} \right) \rhd \mathrm{A}_2$$

$$\oplus \quad [sc_8(s_0,s_1,s_2)] \gg \left(\begin{array}{l} (x|\,\dot{x} = -x) \wedge (x|\,\dot{x} = -x+4) \\ \wedge\,(s_0|\,\dot{s_0} = 0) \wedge (s_1|\,\dot{s_1} = 0) \wedge (s_2|\,\dot{s_2} = 0) \end{array} \right) \rhd \mathrm{A}_2$$

$$\oplus \quad [sc_9(s_0,s_1,s_2)] \gg \left(\; (x|\,\dot{x} = -x) \wedge (s_0|\,\dot{s_0} = 0) \wedge (s_1|\,\dot{s_1} = 0) \wedge (s_2|\,\dot{s_2} = 0) \; \right) \rhd \mathrm{A}_2$$

Then, the encapsulation of Z_0 is calculated (that is the encapsulation of HeaterOn $\|$ Thermostat). After this step, the specification is $\langle\, \mathrm{X8} \,|\, \{\mathrm{X8}, \mathrm{A}_3\}\,\rangle$ where $\mathrm{encap}_{\mathrm{Z}_0}$ is the fresh recursion variable that results from the elimination of encapsulation step.

$$\begin{array}{lcl} \mathrm{X8} & \approx & \left| \left[\begin{array}{c|l} s_0, s_1, s_2, s_3 & \left(\begin{array}{l} [s_0 \,|\, s_0^+ = \mathit{push}(\mathrm{HeaterOn}, \emptyset)] \sim \\ [s_1 \,|\, s_1^+ = \mathit{push}(\mathrm{Thermostat}, \emptyset)] \sim \\ [s_2 \,|\, s_2^+ = \mathit{push}(\mathrm{Z}_0, \emptyset)] \sim \\ [s_3 \,|\, s_3^+ = \mathit{push}(\mathrm{encap}_{\mathrm{Z}_0}, \emptyset)] \end{array} \right) \gg \mathrm{A}_3 \end{array} \right] \right| \\ \mathrm{A}_3 & \approx & [sc_0(s_0,s_1,s_2,s_3)] \sim [x = x_{min}] \gg \mathit{On} \odot \mathrm{A}_3 \\ & \oplus & [sc_1(s_0,s_1,s_2,s_3)] \sim [x = x_{max}] \gg \mathit{Off} \odot \mathrm{A}_3 \\ & \oplus & [sc_2(s_0,s_1,s_2,s_3)] \gg \left(\begin{array}{l} (x|\,\dot{x} = -x) \wedge (x|\,\dot{x} = -x+4) \\ \wedge\,(s_0|\,\dot{s_0} = 0) \wedge (s_1|\,\dot{s_1} = 0) \\ \wedge\,(s_2|\,\dot{s_2} = 0) \wedge (s_3|\,\dot{s_3} = 0) \end{array} \right) \rhd \mathrm{A}_3 \\ & \oplus & [sc_3(s_0,s_1,s_2,s_3)] \gg \left(\begin{array}{l} (x|\,\dot{x} = -x) \wedge (x_{min} \leq x \leq x_{max}) \\ \wedge\,(s_0|\,\dot{s_0} = 0) \wedge (s_1|\,\dot{s_1} = 0) \\ \wedge\,(s_2|\,\dot{s_2} = 0) \wedge (s_3|\,\dot{s_3} = 0) \end{array} \right) \rhd \mathrm{A}_3 \\ & \oplus & [sc_4(s_0,s_1,s_2,s_3)] \gg \left(\begin{array}{l} (x|\,\dot{x} = -x+4) \\ \wedge\,(x_{min} \leq x \leq x_{max}) \\ \wedge\,(s_0|\,\dot{s_0} = 0) \wedge (s_1|\,\dot{s_1} = 0) \\ \wedge\,(s_2|\,\dot{s_2} = 0) \wedge (s_3|\,\dot{s_3} = 0) \end{array} \right) \rhd \mathrm{A}_3 \end{array}$$

Finally, the abstraction in X8 is pulled into the initial term, which gives the linear specification:

$$\langle\, [\![\; s_0, s_1, s_2, s_3 \;|\; i \gg \mathrm{A}_3 \;]\!] \,|\, \{\mathrm{A}_3\}\,\rangle$$

where

$$i = \begin{array}{l} [s_0 \,|\, s_0^+ = \mathit{push}(\mathrm{HeaterOn}, \emptyset)] \sim [s_1 \,|\, s_1^+ = \mathit{push}(\mathrm{Thermostat}, \emptyset)] \sim \\ [s_2 \,|\, s_2^+ = \mathit{push}(\mathrm{Z}_0, \emptyset)] \sim [s_3 \,|\, s_3^+ = \mathit{push}(\mathrm{encap}_{\mathrm{Z}_0}, \emptyset)] \end{array}$$

7 Conclusions and Future Work

We presented a linearization algorithm for the hybrid process algebra HyPA, proved its correctness and presented several optimizations. Our linearization algorithm transforms a $\mathrm{HyPA}_{\mathrm{lin}}$ specification into an equivalent linear recursive specification. Furthermore, we introduced an abstraction operator for HyPA and gave several useful axioms.

The main advantage of linear recursive specifications is that it becomes fairly straightforward to generate the state space from them. Furthermore, a weaker notion of equivalence can be used on linear specifications, which enables the use of certain analysis techniques that cannot be used on normal HyPA specifications. Finally, linear recursive specifications are a convenient form for storage and manipulation by tools.

We have implemented our algorithm in an experimental tool. This tool showed that application of the algorithm to real-world specifications is still problematic, because the re-initialization clauses

on stack variables are very large in the resulting linear specification. Therefore, the most pressing issue currently is optimization of the size of these re-initialization clauses. This optimization would be a very interesting and useful subject of future research, because we feel that this is the last step to enable the linearization of real-world specifications.

Moreover, there are still a number of restrictions on the input specifications because of fundamental difficulties. First, the parallel composition is restricted in such a way that there is no recursion over the parallel composition, as in $X \approx X \parallel Y$ for instance. Second, the abstraction operator is not allowed in the $\mathrm{HyPA}_{\mathrm{lin}}$ form, because it is not possible to eliminate abstraction of open terms from recursive equations. Third, the empty process (ϵ) cannot be used, because it leads to some problems in the transformations. Finally, only single flow clauses can be disrupted. Relaxing these restrictions, especially on the use of recursion, is an interesting topic for future work as well.

References

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

[2] 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.

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

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

[5] P.J.L. Cuijpers and M.A. Reniers. Hybrid process algebra. Technical Report CSR 03-07, TU/e, Eindhoven, Netherlands, 2003.

[6] P.J.L. Cuijpers and M.A. Reniers. Action and predicate safety of hybrid processes. Technical Report CSR 04-10, TU/e, Eindhoven, Netherlands, 2004.

[7] P.J.L. Cuijpers and M.A. Reniers. Hybrid process algebra. *Journal of Logic and Algebraic Programming*, 2004. In press.

[8] J.F. Groote and A. Ponse. The syntax and semantics of μCRL. In A. Ponse, C. Verhoef, and S.F.M. van Vlijmen, editors, *ACP: Algebra of Communicating Processes*, Workshops in Computing, pages 26–62, Utrecht, 1995. Springer-Verlag.

[9] J.F. Groote, A. Ponse, and Y.S. Usenko. Linearization in parallel pCRL. *JLAP*, 48(1-2):39–70, 2001.

[10] 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.

[11] N. Lynch, R. Segala, and F. Vaandrager. Hybrid I/O automata. *Information and Computation*, 185(1):105–157, 2003.

[12] M.R. Mousavi, M.A. Reniers, and J.F. Groote. Congruence for SOS with data. In *Proceedings of Nineteenth Annual IEEE Symposium on Logic in Computer Science (LICS'04)*, pages 302–313, Turku, Finland, 2004. IEEE Computer Society Press.

[13] X. Nicollin, J. Sifakis, and S. Yovine. From ATP to timed graphs and hybrid systems. *Acta Informatica*, 30(2):181–202, 1993.

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

[15] 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. In S. Engell, H. Gueguen, and J. Zaytoon, editors, *IFAC Conference on Analysis and Design of Hybrid Systems (ADHS03)*, pages 235–240, Saint-Malo, France, June 2003.

[16] Y.S. Usenko. *Linearization in μCRL*. PhD thesis, Technische Universiteit Eindhoven (TU/e), 2002.

[17] 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.

A Soundness of Abstraction Axioms

A.1 The Axiom: $[\![V \mid x]\!] \oplus [\![V \mid y]\!] \approx [\![V \mid x \oplus y]\!]$

Take $R \subseteq \mathcal{T} \times \mathcal{T}$ to be the relation

$$R = \{([\![V \mid x]\!] \oplus [\![V \mid y]\!], [\![V \mid x \oplus y]\!]) \mid V \subseteq \mathcal{V}_{\mathrm{m}}, x, y \in \mathcal{T}\} \cup \{(x, x) \mid x \in \mathcal{T}\}$$

For $(([\![V \mid x]\!] \oplus [\![V \mid y]\!], [\![V \mid x \oplus y]\!]) \in R$ we have the following cases:

1. $\langle [\![V \mid x]\!] \oplus [\![V \mid y]\!], \mu \rangle \checkmark$, which needs one of the hypotheses
   (a) $\langle [\![V \mid x]\!], \mu \rangle \checkmark$, which needs the hypothesis
   $\exists_\nu \langle |[V : \nu \mid x]|, \mu \rangle \checkmark$, which needs the hypothesis
   $\langle x, m_V(\mu, \nu) \rangle \checkmark$, so
   $\langle x \oplus y, m_V(\mu, \nu) \rangle \checkmark$, so
   $\langle |[V : \nu \mid x \oplus y]|, \mu \rangle \checkmark$, so
   $\langle [\![V \mid x \oplus y]\!], \mu \rangle \checkmark$.
   (b) $\langle [\![V \mid y]\!], \mu \rangle \checkmark$, which is similar to the case $\langle [\![V \mid x]\!], \mu \rangle \checkmark$.
2. $\langle [\![V \mid x \oplus y]\!], \mu \rangle \checkmark$, which needs the hypothesis
   $\exists_\nu \langle |[V : \nu \mid x \oplus y]|, \mu \rangle \checkmark$, which needs the hypothesis
   $\langle x \oplus y, m_V(\mu, \nu) \rangle \checkmark$, which needs one of the hypotheses
   (a) $\langle x, m_V(\mu, \nu) \rangle \checkmark$, so
   $\langle |[V : \nu \mid x]|, \mu \rangle \checkmark$, so
   $\langle [\![V \mid x]\!], \mu \rangle \checkmark$, so
   $\langle [\![V \mid x]\!] \oplus [\![V \mid y]\!], \mu \rangle \checkmark$.
   (b) $\langle y, m_V(\mu, \nu) \rangle \checkmark$, which is similar to the case $\langle x, m_V(\mu, \nu) \rangle \checkmark$.
3. $\langle [\![V \mid x]\!] \oplus [\![V \mid y]\!], \mu \rangle \overset{a,l}{\mapsto} \langle p, \mu' \rangle$, which needs one of the hypotheses
   (a) $\langle [\![V \mid x]\!], \mu \rangle \overset{a,l}{\mapsto} \langle p, \mu' \rangle$, which needs the hypothesis
   $\exists_\nu \langle |[V : \nu \mid x]|, \mu \rangle \overset{a,l}{\mapsto} \langle p, \mu' \rangle$, which needs the hypothesis
   $\exists_{w,p',w'} \langle x, m_V(\mu, \nu) \rangle \overset{a,w}{\mapsto} \langle p', w' \rangle$ with $l = m_V(w, \mu), p = |[V : w' \restriction_V \mid p']|$, $\mu' = m_V(w', \mu)$, so
   $\langle x \oplus y, m_V(\mu, \nu) \rangle \overset{a,w}{\mapsto} \langle p', w' \rangle$, so
   $\langle |[V : \nu \mid x \oplus y]|, \mu \rangle \overset{a, m_V(w,\mu)}{\mapsto} \langle |[V : w' \restriction_V \mid p']|, m_V(w', \mu) \rangle$, so
   $\langle |[V : \nu \mid x \oplus y]|, \mu \rangle \overset{a,l}{\mapsto} \langle p, \mu' \rangle$, so

$\langle\, [\![\ V\ \mid x \oplus y\ ]\!], \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$.
Note that $(p,p) \in R$.

(b) $\langle\, [\![\ V\ \mid y\ ]\!], \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which is similar to the case $\langle\, [\![\ V\ \mid x\ ]\!], \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$.

4. $\langle\, [\![\ V\ \mid x \oplus y\ ]\!], \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
$\exists_\nu \langle\, |[\ V : \nu \mid x \oplus y\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
$\exists_{w,p',w'} \langle\, x \oplus y, m_V(\mu,\nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$ with $l = m_V(w,\mu),\ p = |[\ V : w'\restriction_V\ \mid p'\ ]|,\ \mu' = m_V(w',\mu)$, which needs one of the hypotheses

(a) $\langle\, x, m_V(\mu,\nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$, so
$\langle\, |[\ V : \nu \mid x\ ]|, \mu \,\rangle \overset{a,m_V(w,\mu)}{\mapsto} \langle\, |[\ V : w'\restriction_V\ \mid p'\ ]|, m_V(w',\mu) \,\rangle$, so
$\langle\, |[\ V : \nu \mid x\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, so
$\langle\, [\![\ V\ \mid x\ ]\!], \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$.
Note that $(p,p) \in R$.

(b) $\langle\, y, m_V(\mu,\nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$, which is similar to the case $\langle\, x, m_V(\mu,\nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$.

5. $\langle\, [\![\ V\ \mid x\ ]\!] \oplus [\![\ V\ \mid y\ ]\!], \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$, which needs one of the hypotheses

(a) $\langle\, [\![\ V\ \mid x\ ]\!], \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
$\exists_\nu \langle\, |[\ V : \nu \mid x\ ]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
$\exists_{\sigma,\sigma',p',w'} \langle\, x, m_V(\mu,\nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w' \,\rangle$ with $s = m_V(\sigma,\sigma'),\ p = |[\ V : w'\restriction_V\ \mid p'\ ]|,\ \mu' = m_V(w',\sigma'(\uparrow))$, so
$\langle\, x \oplus y, m_V(\mu,\nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w' \,\rangle$, so
$\forall_{\sigma''} \langle\, |[\ V : \nu \mid x \oplus y\ ]|, \mu \,\rangle \overset{a,m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V\ \mid p'\ ]|, m_V(w',\sigma''(\uparrow)) \,\rangle$.
We take $\sigma'' = \sigma'$, so
$\langle\, |[\ V : \nu \mid x \oplus y\ ]|, \mu \,\rangle \overset{a,s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$, so
$\langle\, [\![\ V\ \mid x \oplus y\ ]\!], \mu \,\rangle \overset{a,s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$.
Note that $(p,p) \in R$.

(b) $\langle\, [\![\ V\ \mid y\ ]\!], \mu \,\rangle \overset{a,s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$, which is similar to the case $\langle\, [\![\ V\ \mid x\ ]\!], \mu \,\rangle \overset{a,s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$.

6. $\langle\, [\![\ V\ \mid x \oplus y\ ]\!], \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
$\exists_\nu \langle\, |[\ V : \nu \mid x \oplus y\ ]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
$\exists_{\sigma,\sigma',p',w'} \langle\, x \oplus y, m_V(\mu,\nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w' \,\rangle$ with $s = m_V(\sigma,\sigma'),\ p = |[\ V : w'\restriction_V\ \mid p'\ ]|,\ \mu' = m_V(w',\sigma'(\uparrow))$, which needs one of the hypotheses

(a) $\langle\, x, m_V(\mu,\nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w' \,\rangle$, so
$\forall_{\sigma''} \langle\, |[\ V : \nu \mid x\ ]|, \mu \,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V\ \mid p'\ ]|, m_V(w',\sigma''(\uparrow)) \,\rangle$.
We take $\sigma'' = \sigma'$, so
$\langle\, |[\ V : \nu \mid x\ ]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$, so
$\langle\, [\![\ V\ \mid x\ ]\!], \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$.
Note that $(p,p) \in R$.

(b) $\langle\, y, m_V(\mu,\nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w' \,\rangle$, which is similar to the case $\langle\, x, m_V(\mu,\nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w' \,\rangle$.

For $(x,x) \in R$ the proof is trivial.

A.2 The Axiom: $[\![\ V\ \mid x\ ]\!] \odot [\![\ V\ \mid y\ ]\!] \approx [\![\ V\ \mid x \odot [V|\, true] \gg y\ ]\!]$

Take $R \subseteq \mathcal{T} \times \mathcal{T}$ to be the relation

$$
\begin{array}{rcl}
R & = & \{(|[\ V : \nu \mid x\ ]| \odot [\![\ V\ \mid y\ ]\!], |[\ V : \nu \mid x \odot [V|\, true] \gg y\ ]|) \\
 & & \mid V \subseteq \mathcal{V}_{\mathrm{m}}, x, y \in \mathcal{T}, \nu \in \mathit{Val}\} \\
 & \cup & \{([\![\ V\ \mid x\ ]\!] \odot [\![\ V\ \mid y\ ]\!], [\![\ V\ \mid x \odot [V|\, true] \gg y\ ]\!]) \mid V \subseteq \mathcal{V}_{\mathrm{m}}, x, y \in \mathcal{T}\} \\
 & \cup & \{(x,x) \mid x \in \mathcal{T}\}
\end{array}
$$

For $(|[\ V : \nu \mid x\ ]| \odot [\![\ V\ \mid y\ ]\!], |[\ V : \nu \mid x \odot [V|\, true] \gg y\ ]|) \in R$ we have the following cases:

1. $\langle\, |[\ V : \nu \mid x\ ]| \odot [\![\ V\ \mid y\ ]\!], \mu \,\rangle \checkmark$, which needs the hypothesis
   $\langle\, |[\ V : \nu \mid x\ ]|, \mu \,\rangle \checkmark$ and $\langle\, [\![\ V\ \mid y\ ]\!], \mu \,\rangle \checkmark$, which needs the hypothesis
   $\exists_{\nu'}\ \langle\, |[\ V : \nu' \mid y\ ]|, \mu \,\rangle \checkmark$, which needs the hypothesis
   $\langle\, x, m_V(\mu,\nu) \,\rangle \checkmark$ and $\langle\, y, m_V(\mu,\nu') \,\rangle \checkmark$.
   $(m_V(\mu,\nu), m_V(\mu,\nu')) \models [V|\, true]$, so
   $\langle\, [V|\, true] \gg y, m_V(\mu,\nu) \,\rangle \checkmark$, so
   $\langle\, x \odot [V|\, true] \gg y, m_V(\mu,\nu) \,\rangle \checkmark$, so
   $\langle\, |[\ V : \nu \mid x \odot [V|\, true] \gg y\ ]|, \mu \,\rangle \checkmark$.
2. $\langle\, |[\ V : \nu \mid x \odot [V|\, true] \gg y\ ]|, \mu \,\rangle \checkmark$, which needs the hypothesis
   $\langle\, x \odot [V|\, true] \gg y, m_V(\mu,\nu) \,\rangle \checkmark$, which needs the hypothesis
   $\langle\, x, m_V(\mu,\nu) \,\rangle \checkmark$ and $\langle\, [V|\, true] \gg y, m_V(\mu,\nu) \,\rangle \checkmark$.
   $\exists_{\nu'}(m_V(\mu,\nu), m_V(\mu,\nu')) \models [V|\, true]$, so
   $\langle\, y, m_V(\mu,\nu') \,\rangle \checkmark$, so
   $\langle\, |[\ V : \nu \mid x\ ]|, \mu \,\rangle \checkmark$ and $\langle\, |[\ V : \nu' \mid y\ ]|, \mu \,\rangle \checkmark$, so
   $\langle\, [\![\ V\ \mid y\ ]\!], \mu \,\rangle \checkmark$, so
   $\langle\, |[\ V : \nu \mid x\ ]| \odot [\![\ V\ \mid y\ ]\!], \mu \,\rangle \checkmark$.
3. $\langle\, |[\ V : \nu \mid x\ ]| \odot [\![\ V\ \mid y\ ]\!], \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs one of the hypotheses

   (a) $\langle\, [\![\ V\ \mid y\ ]\!], \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$ and $\langle\, |[\ V : \nu \mid x\ ]|, \mu \,\rangle \checkmark$, which needs the hypothesis
   $\exists_{\nu'}\ \langle\, |[\ V : \nu' \mid y\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$ and $\langle\, x, m_V(\mu,\nu) \,\rangle \checkmark$, which needs the hypothesis
   $\exists_{w,p',w'}\ \langle\, y, m_V(\mu,\nu') \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$ with $l = m_V(w,\mu),\ p = |[\ V : w'\restriction_V\ \mid p'\ ]|,\ \mu' = m_V(w',\mu)$. $(m_V(\mu,\nu), m_V(\mu,\nu')) \models [V|\, true]$, so
   $\langle\, [V|\, true] \gg y, m_V(\mu,\nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$, so
   $\langle\, x \odot [V|\, true] \gg y, m_V(\mu,\nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$, so
   $\langle\, |[\ V : \nu \mid x \odot [V|\, true] \gg y\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$.
   Note that $(p,p) \in R$.

   (b) $\exists_{p'}\ \langle\, |[\ V : \nu \mid x\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p', \mu' \,\rangle$ with $p = p' \odot [\![\ V\ \mid y\ ]\!]$, which needs the hypothesis
   $\exists_{w,p'',w'}\ \langle\, x, m_V(\mu,\nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p'', w' \,\rangle$ with $l = m_V(w,\mu), p' = |[\ V : w'\restriction_V\ \mid p''\ ]|,\ \mu' = m_V(w',\mu)$, so
   $\langle\, x \odot [V|\, true] \gg y, m_V(\mu,\nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p'' \odot [V|\, true] \gg y, w' \,\rangle$, so
   $\langle\, |[\ V : \nu \mid x \odot [V|\, true] \gg y\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, |[\ V : w'\restriction_V\ \mid p'' \odot [V|\, true] \gg y\ ]|, \mu' \,\rangle$.
   Recall that $p = p' \odot [\![\ V\ \mid y\ ]\!] = |[\ V : w'\restriction_V\ \mid p''\ ]| \odot [\![\ V\ \mid y\ ]\!]$ and note that $(|[\ V : w'\restriction_V\ \mid p''\ ]| \odot [\![\ V\ \mid y\ ]\!], |[\ V : w'\restriction_V\ \mid p'' \odot [V|\, true] \gg y\ ]|) \in R$

4. $\langle\, |[\ V : \nu \mid x \odot [V|\, true] \gg y\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
   $\exists_{w,p',w'}\ \langle\, x \odot [V|\, true] \gg y, m_V(\mu,\nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$ with $l = m_V(w,\mu),\ p = |[\ V : w'\restriction_V \mid p'\ ]|,\ \mu' = m_V(w',\mu)$, which needs one of the hypotheses

   (a) $\langle\, x, m_V(\mu,\nu) \,\rangle \checkmark$ and $\langle\, [V|\, true] \gg y, m_V(\mu,\nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$, so
   $\langle\, |[\ V : \nu \mid x\ ]|, \mu \,\rangle \checkmark$.
   $\exists_{\nu'}(m_V(\mu,\nu), m_V(\mu,\nu')) \models [V|\, true]$, so
   $\langle\, y, m_V(\mu,\nu') \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$, so
   $\langle\, |[\ V : \nu' \mid y\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, so
   $\langle\, [\![\ V\ \mid y\ ]\!], \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, so
   $\langle\, |[\ V : \nu \mid x\ ]| \odot [\![\ V\ \mid y\ ]\!], \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$.
   Note that $(p,p) \in R$.

   (b) $\exists_{p''}\ \langle\, x, m_V(\mu,\nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p'', w' \,\rangle$ with $p' = p'' \odot [V|\, true] \gg y$, so
   $\langle\, |[\ V : \nu \mid x\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, |[\ V : w'\restriction_V\ \mid p''\ ]|, \mu' \,\rangle$, so
   $\langle\, |[\ V : \nu \mid x\ ]| \odot [\![\ V\ \mid y\ ]\!], \mu \,\rangle \overset{a,l}{\mapsto} \langle\, |[\ V : w'\restriction_V\ \mid p''\ ]| \odot [\![\ V\ \mid y\ ]\!], \mu' \,\rangle$.

Recall that $p = |[\ V : w'\restriction_V \ \mid p'\ ]| = |[\ V : w'\restriction_V \ \mid p'' \odot [V\,|true] \gg y\ ]|$ and note that $(|[\ V : w'\restriction_V \ \mid p''\ ]| \odot [\![\ V \ \mid y\ ]\!], |[\ V : w'\restriction_V \ \mid p'' \odot [V\,|true] \gg y\ ]|) \in R$

5. $\langle\, |[\ V : \nu \mid x\ ]| \odot [\![\ V \ \mid y\ ]\!], \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs one of the hypotheses

   (a) $\langle\, [\![\ V \ \mid y\ ]\!], \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$ and $\langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle \checkmark$, which needs the hypothesis
   $\exists_{\nu'}\, \langle\, |[\ V : \nu' \mid y\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$ and $\langle\, x, m_V(\mu, \nu)\,\rangle \checkmark$, which needs the hypothesis
   $\exists_{\sigma,\sigma',p',w'}\, \langle\, y, m_V(\mu, \nu')\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$ with $s = m_V(\sigma, \sigma')$, $p = |[\ V : w'\restriction_V \ \mid p'\ ]|$, $\mu' = m_V(w', \sigma'(\uparrow))$.
   $(m_V(\mu, \nu), m_V(\mu, \nu')) \models [V\,|true]$, so
   $\langle\, [V\,|true] \gg y, m_V(\mu, \nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$, so
   $\langle\, x \odot [V\,|true] \gg y, m_V(\mu, \nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$, so
   $\forall_{\sigma''}\, \langle\, |[\ V : \nu \mid x \odot [V\,|true] \gg y\ ]|, \mu\,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, p, m_V(w', \sigma''(\uparrow))\,\rangle$.
   We take $\sigma'' = \sigma'$, so
   $\langle\, |[\ V : \nu \mid x \odot [V\,|true] \gg y\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$.
   Note that $(p, p) \in R$.

   (b) $\exists_{p'}\, \langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p', \mu'\,\rangle$ with $p = p' \odot [\![\ V \ \mid y\ ]\!]$, which needs the hypothesis
   $\exists_{\sigma,\sigma',p'',w'}\, \langle\, x, m_V(\mu, \nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'', w'\,\rangle$ with $s = m_V(\sigma, \sigma')$, $p' = |[\ V : w'\restriction_V \ \mid p''\ ]|$, $\mu' = m_V(w', \sigma'(\uparrow))$, so
   $\langle\, x \odot [V\,|true] \gg y, m_V(\mu, \nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'' \odot [V\,|true] \gg y, w'\,\rangle$, so
   $\forall_{\sigma''}\, \langle\, |[\ V : \nu \mid x \odot [V\,|true] \gg y\ ]|, \mu\,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \ \mid p'' \odot [V\,|true] \gg y\ ]|, m_V(w', \sigma''(\uparrow))\,\rangle$.
   We take $\sigma'' = \sigma'$, so
   $\langle\, |[\ V : \nu \mid x \odot [V\,|true] \gg y\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \ \mid p'' \odot [V\,|true] \gg y\ ]|, \mu'\,\rangle$.
   Recall that $p = p' \odot [\![\ V \ \mid y\ ]\!] = |[\ V : w'\restriction_V \ \mid p''\ ]| \odot [\![\ V \ \mid y\ ]\!]$ and note that $(|[\ V : w'\restriction_V \ \mid p''\ ]| \odot [\![\ V \ \mid y\ ]\!], |[\ V : w'\restriction_V \ \mid p'' \odot [V\,|true] \gg y\ ]|) \in R$

6. $\langle\, |[\ V : \nu \mid x \odot [V\,|true] \gg y\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
   $\exists_{\sigma,\sigma',p',w'}\, \langle\, x \odot [V\,|true] \gg y, m_V(\mu, \nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$ with $s = m_V(\sigma, \sigma')$, $p = |[\ V : w'\restriction_V \mid p'\ ]|$, $\mu' = m_V(w', \sigma'(\uparrow))$, which needs one of the hypotheses

   (a) $\langle\, x, m_V(\mu, \nu)\,\rangle \checkmark$ and $\langle\, [V\,|true] \gg y, m_V(\mu, \nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$, so
   $\langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle \checkmark$ and $\exists_{\nu'}(m_V(\mu, \nu), m_V(\mu, \nu')) \models [V\,|true]$, so
   $\langle\, y, m_V(\mu, \nu')\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$, so
   $\forall_{\sigma''}\, \langle\, |[\ V : \nu' \mid y\ ]|, \mu\,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, p, m_V(w', \sigma''(\uparrow)\,\rangle$.
   We take $\sigma'' = \sigma'$, so
   $\langle\, |[\ V : \nu' \mid y\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, so
   $\langle\, [\![\ V \ \mid y\ ]\!], \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, so
   $\langle\, |[\ V : \nu \mid x\ ]| \odot [\![\ V \ \mid y\ ]\!], \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$.
   Note that $(p, p) \in R$.

   (b) $\exists_{p''}\, \langle\, x, m_V(\mu, \nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'', w'\,\rangle$ with $p' = p'' \odot [V\,|true] \gg y$, so
   $\forall_{\sigma''}\, \langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \ \mid p''\ ]|, m_V(w', \sigma''(\uparrow)\,\rangle$, so
   We take $\sigma'' = \sigma'$, so
   $\langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \ \mid p''\ ]|, \mu'\,\rangle$, so
   $\langle\, |[\ V : \nu \mid x\ ]| \odot [\![\ V \ \mid y\ ]\!], \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \ \mid p''\ ]| \odot [\![\ V \ \mid y\ ]\!], \mu'\,\rangle$.
   Recall that $p = |[\ V : w'\restriction_V \ \mid p'\ ]| = |[\ V : w'\restriction_V \ \mid p'' \odot [V\,|true] \gg y\ ]|$ and note that $(|[\ V : w'\restriction_V \ \mid p''\ ]| \odot [\![\ V \ \mid y\ ]\!], |[\ V : w'\restriction_V \ \mid p'' \odot [V\,|true] \gg y\ ]|) \in R$

For $([\![\ V \ \mid x\ ]\!] \odot [\![\ V \ \mid y\ ]\!], [\![\ V \ \mid x \odot [V|\,true] \gg y\ ]\!]) \in R$ we have the following cases:

1. $\langle\, [\![\ V \ \mid x\ ]\!] \odot [\![\ V \ \mid y\ ]\!], \mu\,\rangle \checkmark$, which needs the hypothesis
   $\langle\, [\![\ V \ \mid x\ ]\!], \mu\,\rangle \checkmark$ and $\langle\, [\![\ V \ \mid x\ ]\!], \mu\,\rangle \checkmark$, which needs the hypothesis

$\exists_\nu \langle\, |[\, V : \nu \mid x \,]|, \mu \,\rangle \checkmark$ and $\exists_{\nu'} \langle\, |[\, V : \nu' \mid x \,]|, \mu \,\rangle \checkmark$, which needs the hypothesis
$\langle\, x, m_V(\mu, \nu) \,\rangle \checkmark$ and $\langle\, y, m_V(\mu, \nu') \,\rangle \checkmark$.
$(m_V(\mu, \nu), m_V(\mu, \nu')) \models [V \,|true\,]$, so
$\langle\, [V \,|true\,] \gg y, m_V(\mu, \nu) \,\rangle \checkmark$, so
$\langle\, x \odot [V \,|true\,] \gg y, m_V(\mu, \nu) \,\rangle \checkmark$, so
$\langle\, |[\, V : \nu \mid x \odot [V \,|true\,] \gg y \,]|, \mu \,\rangle \checkmark$, so
$\langle\, [\![\, V \mid x \odot [V \,|true\,] \gg y \,]\!], \mu \,\rangle \checkmark$.

2. $\langle\, [\![\, V \mid x \odot [V \,|true\,] \gg y \,]\!], \mu \,\rangle \checkmark$, which needs the hypothesis
$\exists_\nu \langle\, |[\, V : \nu \mid x \odot [V \,|true\,] \gg y \,]|, \mu \,\rangle \checkmark$, which needs the hypothesis
$\langle\, x \odot [V \,|true\,] \gg y, m_V(\mu, \nu) \,\rangle \checkmark$, which needs the hypothesis
$\langle\, x, m_V(\mu, \nu) \,\rangle \checkmark$ and $\langle\, [V \,|true\,] \gg y, m_V(\mu, \nu) \,\rangle \checkmark$.
$(m_V(\mu, \nu), m_V(\mu, \nu')) \models [V \,|true\,]$, so
$\langle\, y, m_V(\mu, \nu') \,\rangle \checkmark$, so
$\langle\, |[\, V : \nu \mid x \,]|, \mu \,\rangle \checkmark$ and $\langle\, |[\, V : \nu' \mid y \,]|, \mu \,\rangle \checkmark$, so
$\langle\, [\![\, V \mid x \,]\!], \mu \,\rangle \checkmark$ and $\langle\, [\![\, V \mid y \,]\!], \mu \,\rangle \checkmark$, so
$\langle\, [\![\, V \mid x \,]\!] \odot [\![\, V \mid y \,]\!], \mu \,\rangle \checkmark$.

3. $\langle\, [\![\, V \mid x \,]\!] \odot [\![\, V \mid y \,]\!], \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs one of the hypotheses

   (a) $\langle\, [\![\, V \mid y \,]\!], \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$ and $\langle\, [\![\, V \mid x \,]\!], \mu \,\rangle \checkmark$, which needs the hypothesis
   $\exists_\nu \langle\, |[\, V : \nu \mid x \,]|, \mu \,\rangle \checkmark$,
   which is now similar to case 3a in the proof for $(|[\, V : \nu \mid x \,]| \odot [\![\, V \mid y \,]\!], |[\, V : \nu \mid x \odot [V| \,true] \gg y \,]|) \in R$.

   (b) $\exists_{p'} \langle\, [\![\, V \mid x \,]\!], \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p', \mu' \,\rangle$ with p = p' $\odot [\![\, V \mid y \,]\!]$, which needs the hypothesis
   $\exists_\nu \langle\, |[\, V : \nu \mid x \,]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p', \mu' \,\rangle$,
   which is now similar to case 3b in the proof for $(|[\, V : \nu \mid x \,]| \odot [\![\, V \mid y \,]\!], |[\, V : \nu \mid x \odot [V| \,true] \gg y \,]|) \in R$.

4. $\langle\, [\![\, V \mid x \odot [V \,|true\,] \gg y \,]\!], \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
$\exists_\nu \langle\, |[\, V : \nu \mid x \odot [V \,|true\,] \gg y \,]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$,
which is now similar to case 4 in the proof for $(|[\, V : \nu \mid x \,]| \odot [\![\, V \mid y \,]\!], |[\, V : \nu \mid x \odot [V| \,true] \gg y \,]|) \in R$.

5. $\langle\, [\![\, V \mid x \,]\!] \odot [\![\, V \mid y \,]\!], \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$, which needs one of the hypotheses

   (a) $\langle\, [\![\, V \mid y \,]\!], \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$ and $\langle\, [\![\, V \mid x \,]\!], \mu \,\rangle \checkmark$, which needs the hypothesis
   $\exists_\nu \langle\, |[\, V : \nu \mid x \,]|, \mu \,\rangle \checkmark$,
   which is now similar to case 5a in the proof for $(|[\, V : \nu \mid x \,]| \odot [\![\, V \mid y \,]\!], |[\, V : \nu \mid x \odot [V| \,true] \gg y \,]|) \in R$.

   (b) $\exists_{p'} \langle\, [\![\, V \mid x \,]\!], \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p', \mu' \,\rangle$ with p = p' $\odot [\![\, V \mid y \,]\!]$, which needs the hypothesis
   $\exists_\nu \langle\, |[\, V : \nu \mid x \,]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p', \mu' \,\rangle$,
   which is now similar to case 5b in the proof for $(|[\, V : \nu \mid x \,]| \odot [\![\, V \mid y \,]\!], |[\, V : \nu \mid x \odot [V| \,true] \gg y \,]|) \in R$.

6. $\langle\, [\![\, V \mid x \odot [V \,|true\,] \gg y \,]\!], \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
$\exists_\nu \langle\, |[\, V : \nu \mid x \odot [V \,|true\,] \gg y \,]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$,
which is now similar to case 6 in the proof for $(|[\, V : \nu \mid x \,]| \odot [\![\, V \mid y \,]\!], |[\, V : \nu \mid x \odot [V| \,true] \gg y \,]|) \in R$.

For $(x, x) \in R$ the proof is trivial.

A.3 The Axiom: $[\![\ V\ \mid\ x\ ]\!] \rhd [\![\ V\ \mid\ y\ ]\!] \approx [\![\ V\ \mid\ x \rhd [V|\,true] \gg y\ ]\!]$

Take $R \subseteq \mathcal{T} \times \mathcal{T}$ to be the relation

$$\begin{array}{rcl} R & = & \{(|[\ V:\nu \mid x\ ]| \blacktriangleright [\![\ V\ \mid\ y\ ]\!], |[\ V:\nu \mid x \blacktriangleright [V|\,true] \gg y\ ]|) \\ & & \mid V \subseteq \mathcal{V}_{\mathrm{m}}, x, y \in \mathcal{T}, \nu \in \mathit{Val}\} \\ & \cup & \{([\![\ V\ \mid\ x\ ]\!] \rhd [\![\ V\ \mid\ y\ ]\!], [\![\ V\ \mid\ x \rhd [V|\,true] \gg y\ ]\!]) \mid V \subseteq \mathcal{V}_{\mathrm{m}}, x, y \in \mathcal{T}\} \\ & \cup & \{(x,x) \mid x \in \mathcal{T}\} \end{array}$$

For $(|[\ V:\nu \mid x\ ]| \blacktriangleright [\![\ V\ \mid\ y\ ]\!], |[\ V:\nu \mid x \blacktriangleright [V|\,true] \gg y\ ]|) \in R$ we have the following cases:

1. $\langle\, |[\ V:\nu \mid x\ ]| \blacktriangleright [\![\ V\ \mid\ y\ ]\!], \mu \,\rangle \checkmark$, which needs one of the hypotheses
   (a) $\langle\, |[\ V:\nu \mid x\ ]|, \mu \,\rangle \checkmark$, which needs the hypothesis
   $\langle\, x, m_V(\mu,\nu) \,\rangle \checkmark$, so
   $\langle\, x \blacktriangleright [V|\,true] \gg y, m_V(\mu,\nu) \,\rangle \checkmark$, so
   $\langle\, |[\ V:\nu \mid x \blacktriangleright [V|\,true] \gg y\ ]|, \mu \,\rangle \checkmark$.
   (b) $\langle\, [\![\ V\ \mid\ y\ ]\!], \mu \,\rangle \checkmark$, which needs the hypothesis
   $\langle\, |[\ V:\nu' \mid y\ ]|, \mu \,\rangle \checkmark$, which needs the hypothesis
   $\langle\, y, m_V(\mu,\nu') \,\rangle \checkmark$
   $(m_V(\mu,\nu), m_V(\mu,\nu')) \models [V|\,true]$ for any ν, ν' , so
   $\langle\, [V|\,true] \gg y, m_V(\mu,\nu) \,\rangle \checkmark$, so
   $\langle\, x \blacktriangleright [V|\,true] \gg y, m_V(\mu,\nu) \,\rangle \checkmark$, so
   $\langle\, |[\ V:\nu \mid x \blacktriangleright [V|\,true] \gg y\ ]|, \mu \,\rangle \checkmark$.
2. $\langle\, |[\ V:\nu \mid x \blacktriangleright [V|\,true] \gg y\ ]|, \mu \,\rangle \checkmark$, which needs the hypothesis
   $\langle\, x \blacktriangleright [V|\,true] \gg y, m_V(\mu,\nu) \,\rangle \checkmark$, which needs one of the hypotheses
   (a) $\langle\, x, m_V(\mu,\nu) \,\rangle \checkmark$, so
   $\langle\, |[\ V:\nu \mid x\ ]|, \mu \,\rangle \checkmark$, so
   $\langle\, |[\ V:\nu \mid x\ ]| \blacktriangleright [\![\ V\ \mid\ y\ ]\!], \mu \,\rangle \checkmark$.
   (b) $\langle\, [V|\,true] \gg y, m_V(\mu,\nu) \,\rangle \checkmark$, which needs the hypothesis
   $\exists_{\mu''}(m_V(\mu,\nu), \mu'') \models [V|\,true]$ and $\langle\, y, \mu'' \,\rangle \checkmark$.
   $(m_V(\mu,\nu), m_V(\mu,\nu')) \models [V|\,true]$ for any ν' , so
   we can take $\mu'' = m_V(\mu,\nu')$, so
   $\langle\, y, m_V(\mu,\nu') \,\rangle \checkmark$, so
   $\langle\, |[\ V:\nu' \mid y\ ]|, \mu \,\rangle \checkmark$, so
   $\langle\, [\![\ V\ \mid\ y\ ]\!], \mu \,\rangle \checkmark$, so
   $\langle\, |[\ V:\nu \mid x\ ]| \blacktriangleright [\![\ V\ \mid\ y\ ]\!], \mu \,\rangle \checkmark$.
3. $\langle\, |[\ V:\nu \mid x\ ]| \blacktriangleright [\![\ V\ \mid\ y\ ]\!], \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs one of the hypotheses
   (a) $\exists_{p'} \langle\, |[\ V:\nu \mid x\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p', \mu' \,\rangle$ with $p = p' \blacktriangleright [\![\ V\ \mid\ y\ ]\!]$, which needs the hypothesis
   $\exists_{w,p'',w'} \langle\, x, m_V(\mu,\nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p'', w' \,\rangle$ with $l = m_V(w,\mu)$, $p' = |[\ V:w'\restriction_V \mid p''\ ]|$, $\mu' = m_V(w',\mu)$, so
   $\langle\, x \blacktriangleright [V|\,true] \gg y, m_V(\mu,\nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p'' \blacktriangleright [V|\,true] \gg y, w' \,\rangle$, so
   $\langle\, |[\ V:\nu \mid x \blacktriangleright [V|\,true] \gg y\ ]|, \mu \,\rangle \overset{a,m_V(w,\mu)}{\mapsto} \langle\, |[\ V:w'\restriction_V \mid p'' \blacktriangleright [V|\,true] \gg y\ ]|, m_V(w',\mu) \,\rangle$, so
   $\langle\, |[\ V:\nu \mid x \blacktriangleright [V|\,true] \gg y\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, |[\ V:w'\restriction_V \mid p'' \blacktriangleright [V|\,true] \gg y\ ]|, \mu' \,\rangle$.
   Recall that $p = p' \blacktriangleright [\![\ V\ \mid\ y\ ]\!] = |[\ V:w'\restriction_V \mid p''\ ]| \blacktriangleright [\![\ V\ \mid\ y\ ]\!]$ and note that $(|[\ V:w'\restriction_V \mid p''\ ]| \blacktriangleright [\![\ V\ \mid\ y\ ]\!], |[\ V:w'\restriction_V \mid p'' \blacktriangleright [V|\,true] \gg y\ ]|) \in R$
   (b) $\langle\, [\![\ V\ \mid\ y\ ]\!], \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
   $\langle\, |[\ V:\nu' \mid y\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
   $\exists_{w,p',w'} \langle\, y, m_V(\mu,\nu') \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$ with $l = m_V(w,\mu)$, $p = |[\ V:w'\restriction_V \mid p'\ ]|$, $\mu' = m_V(w',\mu)$.

$(m_V(\mu,\nu), m_V(\mu,\nu')) \models [V\,|true\,]$ for any ν , so
$\langle\, [V\,|true\,] \gg y, m_V(\mu,\nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p', w'\,\rangle$, so
$\langle\, x \blacktriangleright [V\,|true\,] \gg y, m_V(\mu,\nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p', w'\,\rangle$, so
$\langle\, |[\ V : \nu \mid x \blacktriangleright [V\,|true\,] \gg y\ ]|, \mu\,\rangle \overset{a,m_V(w,\mu)}{\mapsto} \langle\, |[\ V : w'\restriction_V \mid p'\ ]|, m_V(w',\mu)\,\rangle$, so
$\langle\, |[\ V : \nu \mid x \blacktriangleright [V\,|true\,] \gg y\ ]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, |[\ V : w'\restriction_V \mid p'\ ]|, \mu'\,\rangle$.
Note that $(|[\ V : w'\restriction_V \mid p'\ ]|, |[\ V : w'\restriction_V \mid p'\ ]|) \in R$.

4. $\langle\, |[\ V : \nu \mid x \blacktriangleright [V\,|true\,] \gg y\ ]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
$\exists_{w,p',w'}\, \langle\, x \blacktriangleright [V\,|true\,] \gg y, m_V(\mu,\nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p', w'\,\rangle$ with $l = m_V(w,\mu),\ p = |[\ V : w'\restriction_V \mid p'\ ]|,\ \mu' = m_V(w',\mu)$, which needs one of the hypotheses

   (a) $\exists_{p''}\, \langle\, x, m_V(\mu,\nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p'', w'\,\rangle$ with $p' = p'' \blacktriangleright [V\,|true\,] \gg y$, so
   $\langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{a,m_V(w,\mu)}{\mapsto} \langle\, |[\ V : w'\restriction_V \mid p''\ ]|, m_V(w',\mu)\,\rangle$, so
   $\langle\, |[\ V : \nu \mid x\ ]| \blacktriangleright [\![\ V \mid y\ ]\!], \mu\,\rangle \overset{a,l}{\mapsto} \langle\, |[\ V : w'\restriction_V \mid p''\ ]| \blacktriangleright [\![\ V \mid y\ ]\!], \mu'\,\rangle$.
   Recall that $p = |[\ V : w'\restriction_V \mid p'\ ]| = |[\ V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true\,] \gg y\ ]|$ and note that $(|[\ V : w'\restriction_V \mid p''\ ]| \blacktriangleright [\![\ V \mid y\ ]\!], |[\ V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true\,] \gg y\ ]|) \in R$
   (b) $\langle\, [V\,|true\,] \gg y, m_V(\mu,\nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p', w'\,\rangle$, which needs the hypothesis
   $\exists_{\mu''}(m_V(\mu,\nu), \mu'') \models [V\,|true\,]$ and $\langle\, y, \mu''\,\rangle \overset{a,w}{\mapsto} \langle\, p', w'\,\rangle$.
   $(m_V(\mu,\nu), m_V(\mu,\nu')) \models [V\,|true\,]$ for any ν' , so
   we can take $\mu'' = m_V(\mu,\nu')$, so
   $\langle\, y, m_V(\mu,\nu')\,\rangle \overset{a,w}{\mapsto} \langle\, p', w'\,\rangle$, so
   $\langle\, |[\ V : \nu' \mid y\ ]|, \mu\,\rangle \overset{a,m_V(w,\mu)}{\mapsto} \langle\, |[\ V : w'\restriction_V \mid p'\ ]|, m_V(w',\mu)\,\rangle$, so
   $\langle\, |[\ V : \nu' \mid y\ ]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, |[\ V : w'\restriction_V \mid p'\ ]|, \mu'\,\rangle$, so
   $\langle\, [\![\ V \mid y\ ]\!], \mu\,\rangle \overset{a,l}{\mapsto} \langle\, |[\ V : w'\restriction_V \mid p'\ ]|, \mu'\,\rangle$, so
   $\langle\, |[\ V : \nu \mid x\ ]| \blacktriangleright [\![\ V \mid y\ ]\!], \mu\,\rangle \overset{a,l}{\mapsto} \langle\, |[\ V : w'\restriction_V \mid p'\ ]|, \mu'\,\rangle$.
   Note that $(|[\ V : w'\restriction_V \mid p'\ ]|, |[\ V : w'\restriction_V \mid p'\ ]|) \in R$.

5. $\langle\, |[\ V : \nu \mid x\ ]| \blacktriangleright [\![\ V \mid y\ ]\!], \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs one of the hypotheses

   (a) $\exists_{p'}\, \langle\, |[\ V : \nu \mid x\,]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p', \mu'\,\rangle$ with $p = p' \blacktriangleright [\![\ V \mid y\ ]\!]$, which needs the hypothesis
   $\exists_{\sigma,\sigma',p'',w'}\, \langle\, x, m_V(\mu,\nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'', w'\,\rangle$ with $s = m_V(\sigma,\sigma'),\ p' = |[\ V : w'\restriction_V \mid p''\ ]|,\ \mu' = m_V(w', \sigma'(\uparrow))$, so
   $\langle\, x \blacktriangleright [V\,|true\,] \gg y, m_V(\mu,\nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'' \blacktriangleright [V\,|true\,] \gg y, w'\,\rangle$, so
   $\forall_{\sigma''}\, \langle\, |[\ V : \nu \mid x \blacktriangleright [V\,|true\,] \gg y\ ]|, \mu\,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true\,] \gg y\ ]|, m_V(w', \sigma''(\uparrow))\,\rangle$.
   We take $\sigma'' = \sigma'$, so
   $\langle\, |[\ V : \nu \mid x \blacktriangleright [V\,|true\,] \gg y\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true\,] \gg y\ ]|, \mu'\,\rangle$.
   Recall that $p = p' \blacktriangleright [\![\ V \mid y\ ]\!] = |[\ V : w'\restriction_V \mid p''\ ]| \blacktriangleright [\![\ V \mid y\ ]\!]$ and note that $(|[\ V : w'\restriction_V \mid p''\,]| \blacktriangleright [\![\ V \mid y\ ]\!], |[\ V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true\,] \gg y\ ]|) \in R$
   (b) $\langle\, [\![\ V \mid y\ ]\!], \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
   $\langle\, |[\ V : \nu' \mid y\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
   $\exists_{\sigma,\sigma',p',w'}\, \langle\, y, m_V(\mu,\nu')\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$ with $s = m_V(\sigma,\sigma'),\ p = |[\ V : w'\restriction_V \mid p'\ ]|,\ \mu' = m_V(w', \sigma'(\uparrow))$.
   $(m_V(\mu,\nu), m_V(\mu,\nu')) \models [V\,|true\,]$ for any ν , so
   $\langle\, [V\,|true\,] \gg y, m_V(\mu,\nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$, so
   $\langle\, x \blacktriangleright [V\,|true\,] \gg y, m_V(\mu,\nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$, so
   $\forall_{\sigma''}\, \langle\, |[\ V : \nu \mid x \blacktriangleright [V\,|true\,] \gg y\ ]|, \mu\,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p'\ ]|, m_V(w', \sigma''(\uparrow))\,\rangle$.
   We take $\sigma'' = \sigma'$, so
   $\langle\, |[\ V : \nu \mid x \blacktriangleright [V\,|true\,] \gg y\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p'\ ]|, \mu'\,\rangle$.
   Note that $(|[\ V : w'\restriction_V \mid p'\ ]|, |[\ V : w'\restriction_V \mid p'\ ]|) \in R$.

6. $\langle\, |[\, V : \nu \mid x \blacktriangleright [V\,|true] \gg y\,]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
   $\exists_{\sigma,\sigma',p',w'}\ \langle\, x \blacktriangleright [V\,|true] \gg y, m_V(\mu,\nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$ with $s = m_V(\sigma,\sigma')$, $p = |[\, V : w'\restriction_V \mid p'\,]|$, $\mu' = m_V(w', \sigma'(\uparrow))$, which needs one of the hypotheses
   (a) $\exists_{p''}\ \langle\, x, m_V(\mu,\nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'', w'\,\rangle$ with $p' = p'' \blacktriangleright [V\,|true] \gg y$, so
   $\forall_{\sigma''}\ \langle\, |[\, V : \nu \mid x\,]|, \mu\,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\, V : w'\restriction_V \mid p''\,]|, m_V(w', \sigma''(\uparrow))\,\rangle$.
   We take $\sigma'' = \sigma'$, so
   $\langle\, |[\, V : \nu \mid x\,]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\, V : w'\restriction_V \mid p''\,]|, \mu'\,\rangle$, so
   $\langle\, |[\, V : \nu \mid x\,]| \blacktriangleright [\![\, V \mid y\,]\!], \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\, V : w'\restriction_V \mid p''\,]| \blacktriangleright [\![\, V \mid y\,]\!], \mu'\,\rangle$.
   Recall that $p = |[\, V : w'\restriction_V \mid p'\,]| = |[\, V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true] \gg y\,]|$ and note that $(|[\, V : w'\restriction_V \mid p''\,]| \blacktriangleright [\![\, V \mid y\,]\!], |[\, V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true] \gg y\,]|) \in R$
   (b) $\langle\, [V\,|true] \gg y, m_V(\mu,\nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$, which needs the hypothesis
   $\exists_{\mu''}(m_V(\mu,\nu), \mu'') \models [V\,|true]$ and $\langle\, y, \mu''\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$.
   $(m_V(\mu,\nu), m_V(\mu,\nu')) \models [V\,|true]$ for any ν' , so
   we can take $\mu'' = m_V(\mu,\nu')$, so
   $\langle\, y, m_V(\mu,\nu')\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$, so
   $\forall_{\sigma''}\ \langle\, |[\, V : \nu' \mid y\,]|, \mu\,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\, V : w'\restriction_V \mid p'\,]|, m_V(w', \sigma''(\uparrow))\,\rangle$.
   We take $\sigma'' = \sigma'$, so
   $\langle\, |[\, V : \nu' \mid y\,]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\, V : w'\restriction_V \mid p'\,]|, \mu'\,\rangle$, so
   $\langle\, [\![\, V \mid y\,]\!], \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\, V : w'\restriction_V \mid p'\,]|, \mu'\,\rangle$, so
   $\langle\, |[\, V : \nu \mid x\,]| \blacktriangleright [\![\, V \mid y\,]\!], \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\, V : w'\restriction_V \mid p'\,]|, \mu'\,\rangle$.
   Note that $(|[\, V : w'\restriction_V \mid p'\,]|, |[\, V : w'\restriction_V \mid p'\,]|) \in R$.

For $([\![\, V \mid x\,]\!] \rhd [\![\, V \mid y\,]\!], [\![\, V \mid x \rhd [V|\,true] \gg y\,]\!]) \in R$ we have the following cases:

1. $\langle\, [\![\, V \mid x\,]\!] \rhd [\![\, V \mid y\,]\!], \mu\,\rangle\checkmark$, which needs the hypothesis
   $\langle\, [\![\, V \mid x\,]\!], \mu\,\rangle\checkmark$, which needs the hypothesis
   $\langle\, |[\, V : \nu \mid x\,]|, \mu\,\rangle\checkmark$ for some ν, which needs the hypothesis
   $\langle\, x, m_V(\mu,\nu)\,\rangle\checkmark$, so
   $\langle\, x \rhd [V\,|true] \gg y, m_V(\mu,\nu)\,\rangle\checkmark$, so
   $\langle\, |[\, V : \nu \mid x \rhd [V\,|true] \gg y\,]|, \mu\,\rangle\checkmark$, so
   $\langle\, [\![\, V \mid x \rhd [V\,|true] \gg y\,]\!], \mu\,\rangle\checkmark$.
2. $\langle\, [\![\, V \mid x \rhd [V\,|true] \gg y\,]\!], \mu\,\rangle\checkmark$ for some ν, which needs the hypothesis
   $\langle\, |[\, V : \nu \mid x \rhd [V\,|true] \gg y\,]|, \mu\,\rangle\checkmark$, which needs the hypothesis
   $\langle\, x \rhd [V\,|true] \gg y, m_V(\mu,\nu)\,\rangle\checkmark$, which needs the hypothesis
   $\langle\, x, m_V(\mu,\nu)\,\rangle\checkmark$, so
   $\langle\, |[\, V : \nu \mid x\,]|, \mu\,\rangle\checkmark$, so
   $\langle\, [\![\, V \mid x\,]\!], \mu\,\rangle\checkmark$, so
   $\langle\, [\![\, V \mid x\,]\!] \rhd [\![\, V \mid y\,]\!], \mu\,\rangle\checkmark$.
3. $\langle\, [\![\, V \mid x\,]\!] \rhd [\![\, V \mid y\,]\!], \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
   $\exists_{p'}\ \langle\, [\![\, V \mid x\,]\!], \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p', \mu'\,\rangle$ with p = p' $\blacktriangleright [\![\, V \mid y\,]\!]$, which needs the hypothesis
   $\exists_{p',\nu}\ \langle\, |[\, V : \nu \mid x\,]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p', \mu'\,\rangle$ with p = p' $\blacktriangleright [\![\, V \mid y\,]\!]$, which needs the hypothesis
   $\exists_{w,p'',w'}\ \langle\, x, m_V(\mu,\nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p'', w'\,\rangle$ with $l = m_V(w,\mu)$, $p' = |[\, V : w'\restriction_V \mid p''\,]|$, $\mu' = m_V(w', \mu)$, so
   $\langle\, x \rhd [V\,|true] \gg y, m_V(\mu,\nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p'' \blacktriangleright [V\,|true] \gg y, w'\,\rangle$, so
   $\langle\, |[\, V : \nu \mid x \rhd [V\,|true] \gg y\,]|, \mu\,\rangle \overset{a,m_V(w,\mu)}{\mapsto} \langle\, |[\, V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true] \gg y\,]|, m_V(w', \mu)\,\rangle$, so
   $\langle\, |[\, V : \nu \mid x \rhd [V\,|true] \gg y\,]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, |[\, V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true] \gg y\,]|, \mu'\,\rangle$, so
   $\langle\, [\![\, V \mid x \rhd [V\,|true] \gg y\,]\!], \mu\,\rangle \overset{a,l}{\mapsto} \langle\, |[\, V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true] \gg y\,]|, \mu'\,\rangle$.
   Recall that $p = p' \blacktriangleright [\![\, V \mid y\,]\!] = |[\, V : w'\restriction_V \mid p''\,]| \blacktriangleright [\![\, V \mid y\,]\!]$ and note that $(|[\, V : w'\restriction_V \mid p''\,]| \blacktriangleright [\![\, V \mid y\,]\!], |[\, V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true] \gg y\,]|) \in R$

4. $\langle\, [\![\ V\ \mid x \rhd [V\,|true] \gg y\ ]\!]\,, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
   $\langle\, |[\ V : \nu \mid x \rhd [V\,|true] \gg y\ ]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
   $\exists_{w,p',w'}\ \langle\, x \rhd [V\,|true] \gg y, m_V(\mu,\nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p', w'\,\rangle$ with $l = m_V(w,\mu),\ p = |[\ V : w'\restriction_V \mid p'\ ]|,\ \mu' = m_V(w',\mu)$, which needs the hypothesis
   $\exists_{p''}\ \langle\, x, m_V(\mu,\nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p'', w'\,\rangle$ with $p' = p'' \blacktriangleright [V\,|true] \gg y$, so
   $\langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{a,m_V(w,\mu)}{\mapsto} \langle\, |[\ V : w'\restriction_V \mid p''\ ]|, m_V(w',\mu)\,\rangle$, so
   $\langle\, |[\ V : \nu \mid x\ ]|| \rhd [\![\ V\ \mid y\ ]\!]\,, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, |[\ V : w'\restriction_V \mid p''\ ]|| \blacktriangleright [\![\ V\ \mid y\ ]\!]\,, \mu'\,\rangle$.
   Recall that $p = |[\ V : w'\restriction_V \mid p'\ ]| = |[\ V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true] \gg y\ ]||$ and note that $(|[\ V : w'\restriction_V \mid p''\ ]|| \blacktriangleright [\![\ V\ \mid y\ ]\!]\,, |[\ V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true] \gg y\ ]||) \in R$
5. $\langle\, [\![\ V\ \mid x\ ]\!] \rhd [\![\ V\ \mid y\ ]\!]\,, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
   $\langle\, |[\ V : \nu \mid x\ ]|| \rhd [\![\ V\ \mid y\ ]\!]\,, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
   $\exists_{p'}\ \langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p', \mu'\,\rangle$ with p = p' $\blacktriangleright [\![\ V\ \mid y\ ]\!]$, which needs the hypothesis
   $\exists_{\sigma,\sigma',p'',w'}\ \langle\, x, m_V(\mu,\nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'', w'\,\rangle$ with $s = m_V(\sigma,\sigma'),\ p' = |[\ V : w'\restriction_V \mid p''\ ]|,\ \mu' = m_V(w', \sigma'(\uparrow))$, so
   $\langle\, x \rhd [V\,|true] \gg y, m_V(\mu,\nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'' \blacktriangleright [V\,|true] \gg y, w'\,\rangle$, so
   $\forall_{\sigma''}\ \langle\, |[\ V : \nu \mid x \rhd [V\,|true] \gg y\ ]|, \mu\,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true] \gg y\ ]|, m_V(w', \sigma''(\uparrow))\,\rangle$.
   We take $\sigma'' = \sigma'$, so
   $\langle\, |[\ V : \nu \mid x \rhd [V\,|true] \gg y\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true] \gg y\ ]|, \mu'\,\rangle$, so
   $\langle\, [\![\ V\ \mid x \rhd [V\,|true] \gg y\ ]\!]\,, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true] \gg y\ ]|, \mu'\,\rangle$.
   Recall that $p = p' \blacktriangleright [\![\ V\ \mid y\ ]\!] = |[\ V : w'\restriction_V \mid p''\ ]| \blacktriangleright [\![\ V\ \mid y\ ]\!]$ and note that $(|[\ V : w'\restriction_V \mid p''\ ]| \blacktriangleright [\![\ V\ \mid y\ ]\!]\,, |[\ V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true] \gg y\ ]||) \in R$
6. $\langle\, [\![\ V\ \mid x \rhd [V\,|true] \gg y\ ]\!]\,, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
   $\langle\, |[\ V : \nu \mid x \rhd [V\,|true] \gg y\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
   $\exists_{\sigma,\sigma',p',w'}\ \langle\, x \rhd [V\,|true] \gg y, m_V(\mu,\nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$ with $s = m_V(\sigma,\sigma'),\ p = |[\ V : w'\restriction_V \mid p'\ ]|,\ \mu' = m_V(w', \sigma'(\uparrow))$, which needs the hypothesis
   $\exists_{p''}\ \langle\, x, m_V(\mu,\nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'', w'\,\rangle$ with $p' = p'' \blacktriangleright [V\,|true] \gg y$, so
   $\forall_{\sigma''}\ \langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p''\ ]|, m_V(w', \sigma''(\uparrow))\,\rangle$.
   We take $\sigma'' = \sigma'$, so
   $\langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p''\ ]|, \mu'\,\rangle$, so
   $\langle\, |[\ V : \nu \mid x\ ]|| \rhd [\![\ V\ \mid y\ ]\!]\,, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p''\ ]|| \blacktriangleright [\![\ V\ \mid y\ ]\!]\,, \mu'\,\rangle$, so
   $\langle\, [\![\ V\ \mid x\ ]\!] \rhd [\![\ V\ \mid y\ ]\!]\,, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p''\ ]|| \blacktriangleright [\![\ V\ \mid y\ ]\!]\,, \mu'\,\rangle$.
   Recall that $p = |[\ V : w'\restriction_V \mid p'\ ]| = |[\ V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true] \gg y\ ]||$ and note that $(|[\ V : w'\restriction_V \mid p''\ ]|| \blacktriangleright [\![\ V\ \mid y\ ]\!]\,, |[\ V : w'\restriction_V \mid p'' \blacktriangleright [V\,|true] \gg y\ ]||) \in R$

For $(x,x) \in R$ the proof is trivial.

A.4 The Axiom: $[\![\ V\ \mid \partial_H(x)\ ]\!] \approx \partial_H([\![\ V\ \mid x\ ]\!])$

Take $R \subseteq \mathcal{T} \times \mathcal{T}$ to be the relation

$$\begin{array}{rcl} R & = & \{(|[\ V : \nu \mid \partial_H(x)\ ]|, \partial_H(|[\ V : \nu \mid x\ ]|)) \mid V \subseteq \mathcal{V}_{\mathrm{m}}, H \subseteq \mathcal{A}, x \in \mathcal{T}, \nu \in \mathit{Val}\} \\ & \cup & \{([\![\ V\ \mid \partial_H(x)\ ]\!], \partial_H([\![\ V\ \mid x\ ]\!])) \mid V \subseteq \mathcal{V}_{\mathrm{m}}, H \subseteq \mathcal{A}, x \in \mathcal{T}\} \end{array}$$

For $(|[\ V : \nu \mid \partial_H(x)\ ]|, \partial_H(|[\ V : \nu \mid x\ ]|)) \in R$ we have the following cases:

1. $\langle\, |[\ V : \nu \mid \partial_H(x)\ ]|, \mu\,\rangle \checkmark$, which needs the hypothesis
   $\langle\, \partial_H(x), m_V(\mu,\nu)\,\rangle \checkmark$, which needs the hypothesis
   $\langle\, x, m_V(\mu,\nu)\,\rangle \checkmark$, so

$\langle\, |[\, V : \nu \mid x\,]|, \mu \,\rangle \checkmark$, so
$\langle\, \partial_H\, (|[\, V : \nu \mid x\,]|)\, , \mu \,\rangle \checkmark$.

2. $\langle\, \partial_H\, (|[\, V : \nu \mid x\,]|)\, , \mu \,\rangle \checkmark$, which needs the hypothesis
$\langle\, |[\, V : \nu \mid x\,]|, \mu \,\rangle \checkmark$, which needs the hypothesis
$\langle\, x, m_V(\mu, \nu) \,\rangle \checkmark$, so
$\langle\, \partial_H\, (x)\, , m_V(\mu, \nu) \,\rangle \checkmark$, so
$\langle\, |[\, V : \nu \mid \partial_H\, (x)\,]|, \mu \,\rangle \checkmark$.
3. $\langle\, |[\, V : \nu \mid \partial_H\, (x)\,]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
$\exists_{w,p',w'} \langle\, \partial_H\, (x)\, , m_V(\mu, \nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$ with $l = m_V(w, \mu),\ p = |[\, V : w'{\restriction}_V \mid p'\,]|,\ \mu' = m_V(w', \mu)$, which needs one of the hypotheses

   (a) $a \notin H$ and $\exists_{p''} \langle\, x, m_V(\mu, \nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p'', w' \,\rangle$ with $p' = \partial_H\, (p'')$, so
   $\langle\, |[\, V : \nu \mid x\,]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, |[\, V : w'{\restriction}_V \mid p''\,]|, \mu' \,\rangle$, so
   $\langle\, \partial_H\, (|[\, V : \nu \mid x\,]|)\, , \mu \,\rangle \overset{a,l}{\mapsto} \langle\, \partial_H\, (|[\, V : w'{\restriction}_V \mid p''\,]|)\, , \mu' \,\rangle$.
   Recall that $p = |[\, V : w'{\restriction}_V \mid p'\,]| = |[\, V : w'{\restriction}_V \mid \partial_H\, (p'')\,]|$ and note that $(|[\, V : w'{\restriction}_V \mid \partial_H\, (p'')\,]|, \partial_H\, (|[\, V : w'{\restriction}_V \mid p''\,]|)) \in R$
   (b) $a \in H$: No transition is possible, which contradicts with our hypothesis.

4. $\langle\, \partial_H\, (|[\, V : \nu \mid x\,]|)\, , \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs one of the hypotheses

   (a) $a \notin H$ and $\exists_{p'} \langle\, |[\, V : \nu \mid x\,]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p', \mu' \,\rangle$ with $p = \partial_H\, (p')$, which needs the hypothesis
   $\exists_{w,p'',w'} \langle\, x, m_V(\mu, \nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p'', w' \,\rangle$ with $l = m_V(w, \mu),\ p' = |[\, V : w'{\restriction}_V \mid p''\,]|,\ \mu' = m_V(w', \mu)$, so
   $\langle\, \partial_H\, (x)\, , m_V(\mu, \nu) \,\rangle \overset{a,w}{\mapsto} \langle\, \partial_H\, (p'')\, , w' \,\rangle$, so
   $\langle\, |[\, V : \nu \mid \partial_H\, (x)\,]|, \mu) \,\rangle \overset{a,l}{\mapsto} \langle\, |[\, V : w'{\restriction}_V \mid \partial_H\, (p'')\,]|, \mu' \,\rangle$.
   Recall that $p = \partial_H\, (p') = \partial_H\, (|[\, V : w'{\restriction}_V \mid p''\,]|)$ and note that $(|[\, V : w'{\restriction}_V \mid \partial_H\, (p'')\,]|$, $\partial_H\, (|[\, V : w'{\restriction}_V \mid p''\,]|)) \in R$
   (b) $a \in H$: No transition is possible, which contradicts with our hypothesis.

5. $\langle\, |[\, V : \nu \mid \partial_H\, (x)\,]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
$\exists_{\sigma,\sigma',p',w'} \langle\, \partial_H\, (x)\, , m_V(\mu, \nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w' \,\rangle$ with $s = m_V(\sigma, \sigma'),\ p = |[\, V : w'{\restriction}_V \mid p'\,]|,\ \mu' = m_V(w', \sigma'(\uparrow))$, which needs the hypothesis
$\exists_{p''} \langle\, x, m_V(\mu, \nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'', w' \,\rangle$ with $p' = \partial_H\, (p'')$, so
$\forall_{\sigma''} \langle\, |[\, V : \nu \mid x\,]|, \mu \,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\, V : w'{\restriction}_V \mid p''\,]|, m_V(w', \sigma''(\uparrow)) \,\rangle$.
We take $\sigma'' = \sigma'$, so
$\langle\, |[\, V : \nu \mid x\,]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\, V : w'{\restriction}_V \mid p''\,]|, \mu' \,\rangle$, so
$\langle\, \partial_H\, (|[\, V : \nu \mid x\,]|)\, , \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, \partial_H\, (|[\, V : w'{\restriction}_V \mid p''\,]|)\, , \mu' \,\rangle$.
Recall that $p = |[\, V : w'{\restriction}_V \mid p'\,]| = |[\, V : w'{\restriction}_V \mid \partial_H\, (p'')\,]|$ and note that $(|[\, V : w'{\restriction}_V \mid \partial_H\, (p'')\,]|, \partial_H\, (|[\, V : w'{\restriction}_V \mid p''\,]|)) \in R$
6. $\langle\, \partial_H\, (|[\, V : \nu \mid x\,]|)\, , \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
$\exists_{p'} \langle\, |[\, V : \nu \mid x\,]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p', \mu' \,\rangle$ with p = $\partial_H\, (p')$, which needs the hypothesis
$\exists_{\sigma,\sigma',p'',w'} \langle\, x, m_V(\mu, \nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'', w' \,\rangle$ with $s = m_V(\sigma, \sigma'),\ p' = |[\, V : w'{\restriction}_V \mid p''\,]|,\ \mu' = m_V(w', \sigma'(\uparrow))$, so
$\langle\, \partial_H\, (x)\, , m_V(\mu, \nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, \partial_H\, (p'')\, , w' \,\rangle$, so
$\forall_{\sigma''} \langle\, |[\, V : \nu \mid \partial_H\, (x)\,]|, \mu \,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\, V : w'{\restriction}_V \mid \partial_H\, (p'')\,]|, m_V(w', \sigma'') \,\rangle$.
We take $\sigma'' = \sigma'$, so
$\langle\, |[\, V : \nu \mid \partial_H\, (x)\,]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\, V : w'{\restriction}_V \mid \partial_H\, (p'')\,]|, \mu' \,\rangle$.
Recall that $p = \partial_H\, (p') = \partial_H\, (|[\, V : w'{\restriction}_V \mid p''\,]|)$ and note that $(|[\, V : w'{\restriction}_V \mid \partial_H\, (p'')\,]|$, $\partial_H\, (|[\, V : w'{\restriction}_V \mid p''\,]|)) \in R$

The proof for the case of $(|[\![\, V \mid \partial_H\, (x)\,]\!]\, , \partial_H\, (|[\![\, V \mid x\,]\!]\,)) \in R$ is analogous to the case $(|[\, V : \nu \mid \partial_H\, (x)\,]|, \partial_H\, (|[\, V : \nu \mid x\,]|)) \in R$.

A.5 The Axiom: $[\![\ V\ \mid [\![\ W\ \mid\ x\ ]\!]\ ]\!] \approx [\![\ V \cup W\ \mid\ x\ ]\!]$

Take $R \subseteq \mathcal{T} \times \mathcal{T}$ to be the relation

$$\begin{array}{rcl} R & = & \{(|[\ V : \nu \mid |[\ W : \omega \mid x\ ]|\ ]|, |[\ V \cup W : m_W(\nu,\omega) \mid x\ ]|) \mid V, W \subseteq \mathcal{V}_{\mathrm{m}}, x \in \mathcal{T}, \nu, \omega \in \mathit{Val}\} \\ & \cup & \{([\![\ V\ \mid [\![\ W\ \mid\ x\ ]\!]\ ]\!], [\![\ V \cup W\ \mid\ x\ ]\!]) \mid V, W \subseteq \mathcal{V}_{\mathrm{m}}, x \in \mathcal{T}\} \end{array}$$

For $(|[\ V : \nu \mid |[\ W : \omega \mid x\ ]|\ ]|, |[\ V \cup W : m_W(\nu,\omega) \mid x\ ]|) \in R$ we have the following cases:

1. $\langle\, |[\ V : \nu \mid |[\ W : \omega \mid x\ ]|\ ]|, \mu \,\rangle \checkmark$, which needs the hypothesis
   $\langle\, |[\ W : \omega \mid x\ ]|, m_V(\mu,\nu) \,\rangle \checkmark$, which needs the hypothesis
   $\langle\, x, m_W(m_V(\mu,\nu),\omega) \,\rangle \checkmark$, so by lemma 36
   $\langle\, x, m_{V\cup W}(\mu, m_W(\nu,\omega)) \,\rangle \checkmark$, so
   $\langle\, |[\ V \cup W : m_W(\nu,\omega) \mid x\ ]|, \mu \,\rangle \checkmark$.
2. $\langle\, |[\ V \cup W : m_W(\nu,\omega) \mid x\ ]|, \mu \,\rangle \checkmark$, which needs the hypothesis
   $\langle\, x, m_{V\cup W}(\mu, m_W(\nu,\omega)) \,\rangle \checkmark$, so by lemma 36
   $\langle\, x, m_W(m_V(\mu,\nu),\omega) \,\rangle \checkmark$, so
   $\langle\, |[\ W : \omega \mid x\ ]|, m_V(\mu,\nu) \,\rangle \checkmark$, so
   $\langle\, |[\ V : \nu \mid |[\ W : \omega \mid x\ ]|\ ]|, \mu \,\rangle \checkmark$.
3. $\langle\, |[\ V : \nu \mid |[\ W : \omega \mid x\ ]|\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
   $\exists_{w,p',w'} \langle\, |[\ W : \omega \mid x\ ]|, m_V(\mu,\nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$ with $l = m_V(w,\mu)$, $p = |[\ V : w'\restriction_V \mid p'\ ]|$, $\mu' = m_V(w',\mu)$, which needs the hypothesis
   $\exists_{w'',p'',w'''} \langle\, x, m_W(m_V(\mu,\nu),\omega) \,\rangle \overset{a,w''}{\mapsto} \langle\, p'', w''' \,\rangle$ with $w = m_W(w'', m_V(\mu,\nu))$, $p' = |[\ W : w'''\restriction_W \mid p''\ ]|$, $w' = m_W(w''', m_V(\mu,\nu))$, so by lemma 36
   $\langle\, x, m_{V\cup W}(\mu, m_W(\nu,\omega)) \,\rangle \overset{a,w''}{\mapsto} \langle\, p'', w''' \,\rangle$, so
   $\langle\, |[\ V \cup W : m_W(\nu,\omega) \mid x\ ]|, \mu \,\rangle \overset{a,m_{V\cup W}(w'',\mu)}{\mapsto} \langle\, |[\ V \cup W : w'''\restriction_{V\cup W} \mid p''\ ]|, m_{V\cup W}(w''',\mu) \,\rangle$.
   By lemma 37, $l = m_V(w,\mu) = m_V(m_W(w'', m_V(\mu,\nu)),\mu) = m_{V\cup W}(w'',\mu)$ and similarly, $\mu' = m_{V\cup W}(w''',\mu)$, so
   $\langle\, |[\ V \cup W : m_W(\nu,\omega) \mid x\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, |[\ V \cup W : w'''\restriction_{V\cup W} \mid p''\ ]|, \mu' \,\rangle$, so by lemma 38
   $\langle\, |[\ V \cup W : m_W(\nu,\omega) \mid x\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, |[\ V \cup W : m_W(m_W(w''', m_V(\mu,\nu))\restriction_V, w'''\restriction_W) \mid p''\ ]|, \mu' \,\rangle$, so
   $\langle\, |[\ V \cup W : m_W(\nu,\omega) \mid x\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, |[\ V \cup W : m_W(w'\restriction_V, w'''\restriction_W) \mid p''\ ]|, \mu' \,\rangle$.
   Recall that $p = |[\ V : w'\restriction_V \mid p'\ ]| = |[\ V : w'\restriction_V \mid |[\ W : w'''\restriction_W \mid p''\ ]|\ ]|$ and note that $(|[\ V : w'\restriction_V \mid |[\ W : w'''\restriction_W \mid p''\ ]|\ ]|, |[\ V \cup W : m_W(w'\restriction_V, w'''\restriction_W) \mid p''\ ]|) \in R$
4. $\langle\, |[\ V \cup W : m_W(\nu,\omega) \mid x\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
   $\exists_{w,p',w'} \langle\, x, m_{V\cup W}(\nu, m_W(\nu,\omega)) \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$ with $l = m_{V\cup W}(w,\mu)$, $p = |[\ V \cup W : w'\restriction_{V\cup W} \mid p'\ ]|$, $\mu' = m_{V\cup W}(w',\mu)$, so by lemma 36
   $\langle\, x, m_W(m_V(\mu,\nu),\omega) \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$, so
   $\langle\, |[\ W : \omega \mid x\ ]|, m_V(\mu,\nu) \,\rangle \overset{a,m_W(w,m_V(\mu,\nu))}{\mapsto} \langle\, |[\ W : w'\restriction_W \mid p'\ ]|, m_W(w', m_V(\mu,\nu)) \,\rangle$, so
   $\langle\, |[\ V : \nu \mid |[\ W : \omega \mid x\ ]|\ ]|, \mu \,\rangle \overset{a,m_V(m_W(w,m_V(\mu,\nu)),\mu)}{\mapsto} \langle\, |[\ V : m_W(w', m_V(\mu,\nu))\restriction_V \mid |[\ W : w'\restriction_W \mid p'\ ]|\ ]|, m_V(m_W(w', m_V(\mu,\nu)),\mu) \,\rangle$.
   By lemma 37, $l = m_{V\cup W}(w,\mu) = m_V(m_W(w, m_V(\mu,\nu)),\mu)$ and similarly, $\mu' = m_V(m_W(w', m_V(\mu,\nu)),\mu)$, so
   $\langle\, |[\ V : \nu \mid |[\ W : \omega \mid x\ ]|\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, |[\ V : m_W(w', m_V(\mu,\nu))\restriction_V \mid |[\ W : w'\restriction_W \mid p'\ ]|\ ]|, \mu' \,\rangle$.
   Note that by lemma 38, $p = |[\ V \cup W : w'\restriction_{V\cup W} \mid p'\ ]| = |[\ V \cup W : m_W(m_W(w', m_V(\mu,\nu))\restriction_V, w'\restriction_W) \mid p'\ ]|$ and $(|[\ V : m_W(w', m_V(\mu,\nu))\restriction_V \mid |[\ W : w'\restriction_W \mid p'\ ]|\ ]|, |[\ V \cup W : m_W(m_W(w', m_V(\mu,\nu))\restriction_V, w'\restriction_W) \mid p'\ ]|) \in R$.
5. $\langle\, |[\ V : \nu \mid |[\ W : \omega \mid x\ ]|\ ]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
   $\exists_{\sigma,\sigma',p',w'} \langle\, |[\ W : \omega \mid x\ ]|, m_V(\mu,\nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w' \,\rangle$ with $s = m_V(\sigma,\sigma')$, $p = |[\ V : w'\restriction_V \mid p'\ ]|$, $\mu' = m_V(w', \sigma'(\uparrow))$, which needs the hypothesis

$\exists_{\sigma'',\sigma''',p'',w''}\ \langle\, x, m_W(m_V(\mu,\nu),\omega)\,\rangle \overset{\sigma''}{\rightsquigarrow} \langle\, p'', w''\,\rangle$ with $\sigma = m_W(\sigma'',\sigma''')$, $p' = |[\ W : w''{\restriction_W} \mid p''\ ]|$, $w' = m_W(w'',\sigma'''(\uparrow))$, so by lemma 36
$\langle\, x, m_{V\cup W}(\mu, m_W(\nu,\omega))\,\rangle \overset{\sigma''}{\rightsquigarrow} \langle\, p'', w''\,\rangle$, so
$\forall_{\sigma''''}\ \langle\, |[\ V\cup W : m_W(\nu,\omega) \mid x\ ]|, \mu\,\rangle \overset{m_{V\cup W}(\sigma'',\sigma'''')}{\rightsquigarrow}$
$\langle\, |[\ V\cup W : w''{\restriction_{V\cup W}} \mid p''\ ]|, m_{V\cup W}(w'',\sigma''''(\uparrow))\,\rangle$.
We take $\sigma'''' = m_V(\sigma''',\sigma')$ and $\uparrow dom(\sigma'''') = \uparrow dom(\sigma''') = \uparrow dom(\sigma')$.
By lemma 36, $m_{V\cup W}(\sigma'',\sigma'''') = m_{V\cup W}(\sigma'', m_V(\sigma''',\sigma')) =$
$m_V(m_W(\sigma'',\sigma'''),\sigma') = m_V(\sigma,\sigma') = s$.
By lemma 39, $\sigma''''(\uparrow) = m_V(\sigma'''(\uparrow),\sigma'(\uparrow))$, so by lemma 36
$m_{V\cup W}(w'',\sigma''''(\uparrow)) = m_{V\cup W}(w'', m_V(\sigma'''(\uparrow),\sigma'(\uparrow))) = m_V(m_W(w'',\sigma'''(\uparrow)),\sigma'(\uparrow)) =$
$m_V(w',\sigma'(\uparrow)) = \mu'$, so
$\langle\, |[\ V\cup W : m_W(\nu,\omega) \mid x\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V\cup W : w''{\restriction_{V\cup W}} \mid p''\ ]|, \mu'\,\rangle$, so by lemma 38
$\langle\, |[\ V\cup W : m_W(\nu,\omega) \mid x\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V\cup W : m_W(m_W(w'',\sigma'''(\uparrow)){\restriction_V}, w''{\restriction_W}) \mid p''\ ]|, \mu'\,\rangle$
, so
$\langle\, |[\ V\cup W : m_W(\nu,\omega) \mid x\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V\cup W : m_W(w'{\restriction_V}, w''{\restriction_W}) \mid p''\ ]|, \mu'\,\rangle$.
Recall that $p = |[\ V : w'{\restriction_V} \mid p'\ ]| = |[\ V : w'{\restriction_V} \mid |[\ W : w''{\restriction_W} \mid p''\ ]|\ ]|$ and note that $(|[\ V : w'{\restriction_V} \mid |[\ W : w''{\restriction_W} \mid p''\ ]|\ ]|, |[\ V\cup W : m_W(w'{\restriction_V}, w''{\restriction_W}) \mid p''\ ]|) \in R$

6. $\langle\, |[\ V\cup W : m_W(\nu,\omega) \mid x\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
$\exists_{\sigma,\sigma',p',w'}\ \langle\, x, m_{V\cup W}(\mu, m_W(\nu,\omega))\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$ with $s = m_{V\cup W}(\sigma,\sigma')$, $p = |[\ V\cup W : w'{\restriction_{V\cup W}} \mid p'\ ]|$, $\mu' = m_{V\cup W}(w',\sigma'(\uparrow))$, so by lemma 36
$\langle\, x, m_W(m_V(\mu,\nu),\omega)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$, so
$\forall_{\sigma''}\ \langle\, |[\ W : \omega \mid x\ ]|, m_V(\mu,\nu)\,\rangle \overset{m_W(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\ W : w'{\restriction_W} \mid p'\ ]|, m_W(w',\sigma''(\uparrow))\,\rangle$, so
$\forall_{\sigma'''}\ \langle\, |[\ V : \nu \mid |[\ W : \omega \mid x\ ]|\ ]|, \mu\,\rangle \overset{m_V(m_W(\sigma,\sigma''),\sigma''')}{\rightsquigarrow} \langle\, |[\ V : m_W(w',\sigma''(\uparrow)){\restriction_V} \mid |[\ W : w'{\restriction_W} \mid p'\ ]|\ ]|,$
$m_V(m_W(w',\sigma''(\uparrow)),\sigma'''(\uparrow))\,\rangle$.
We take $\sigma'' = \sigma''' = \sigma'$, so
$m_V(m_W(\sigma,\sigma''),\sigma''') = m_V(m_W(\sigma,\sigma'),\sigma') = m_{V\cup W}(\sigma, m_V(\sigma',\sigma')) = m_{V\cup W}(\sigma,\sigma') = s$ and similarly, $m_V(m_W(w',\sigma''),\sigma''') = \mu'$, so
$\langle\, |[\ V : \nu \mid |[\ W : \omega \mid x\ ]|\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V : m_W(w',\sigma''(\uparrow)){\restriction_V} \mid |[\ W : w'{\restriction_W} \mid p'\ ]|\ ]|, \mu'\,\rangle$.
Note that by lemma 38, $p = |[\ V\cup W : w'{\restriction_{V\cup W}} \mid p'\ ]| = |[\ V\cup W : m_W(m_W(w',\sigma''(\uparrow)){\restriction_V}, w'{\restriction_W}) \mid p'\ ]|$ and $(|[\ V : m_W(w',\sigma''(\uparrow)){\restriction_V} \mid |[\ W : w'{\restriction_W} \mid p'\ ]|\ ]|, |[\ V\cup W : m_W(m_W(w',\sigma''(\uparrow)){\restriction_V}, w'{\restriction_W}) \mid p'\ ]|) \in R$.

The proof for the case of $(\llbracket\ V \mid \llbracket\ W \mid x\ \rrbracket\ \rrbracket, \llbracket\ V\cup W \mid x\ \rrbracket) \in R$ is analogous to the case $(|[\ V : \nu \mid |[\ W : \omega \mid x\ ]|\ ]|, |[\ V\cup W : m_W(\nu,\omega) \mid x\ ]|) \in R$.

Lemma 36 *$m_W(m_V(\mu,\nu),\omega) = m_{V\cup W}(\mu, m_W(\nu,\omega))$, for any $\mu,\nu,\omega \in Val$ and $V, W \subseteq \mathcal{V}_{\mathrm{m}}$.*

Proof We prove that these valuations are equal by proving that they are equal for every $n \in \mathcal{V}_{\mathrm{m}}$ in their domain:

$$
\begin{aligned}
& m_W(m_V(\mu,\nu),\omega)(n) \\
= \ & \begin{cases} m_V(\mu,\nu)(n) & \text{if } n \notin W \\ \omega(n) & \text{if } n \in W \end{cases} \\
= \ & \begin{cases} \mu(n) & \text{if } n \notin V \wedge n \notin W \\ \nu(n) & \text{if } n \in V \wedge n \notin W \\ \omega(n) & \text{if } n \in W \end{cases}
\end{aligned}
$$

$$
\begin{aligned}
&= \begin{cases} \mu(n) & \text{if } n \notin V \cup W \\ \nu(n) & \text{if } n \notin W \wedge n \in V \cup W \\ \omega(n) & \text{if } n \in W \wedge n \in V \cup W \end{cases} \\
&= \begin{cases} \mu(n) & \text{if } n \notin V \cup W \\ m_W(\nu, \omega)(n) & \text{if } n \in V \cup W \end{cases} \\
&= m_{V \cup W}(\mu, m_W(\nu, \omega))(n)
\end{aligned}
$$

⊠

Lemma 37 *$m_V(m_W(\omega, m_V(\mu, \nu)), \mu) = m_{V \cup W}(\omega, \mu)$, for any $\mu, \nu, \omega \in Val$ and $V, W \subseteq \mathcal{V}_{\mathrm{m}}$.*

Proof We prove that these valuations are equal by proving that they are equal for every $n \in \mathcal{V}_{\mathrm{m}}$ in their domain:

$$
\begin{aligned}
& m_V(m_W(\omega, m_V(\mu, \nu)), \mu)(n) \\
&= \begin{cases} m_W(\omega, m_V(\mu, \nu))(n) & \text{if } n \notin V \\ \mu(n) & \text{if } n \in V \end{cases} \\
&= \begin{cases} \omega(n) & \text{if } n \notin V \wedge n \notin W \\ m_V(\mu, \nu)(n) & \text{if } n \notin V \wedge n \in W \\ \mu(n) & \text{if } n \in V \end{cases} \\
&= \begin{cases} \omega(n) & \text{if } n \notin V \wedge n \notin W \\ \mu(n) & \text{if } n \notin V \wedge n \in W \wedge n \notin V \\ \nu(n) & \text{if } n \notin V \wedge n \in W \wedge n \in V \\ \mu(n) & \text{if } n \in V \end{cases} \\
&= \begin{cases} \omega(n) & \text{if } n \notin V \cup W \\ \mu(n) & \text{if } n \in V \cup W \end{cases} \\
&= m_{V \cup W}(\omega, \mu)(n)
\end{aligned}
$$

⊠

Lemma 38 *$m_W(m_W(\omega, \mu)\restriction_V, \omega\restriction_W) = \omega\restriction_{V \cup W}$, for any $\mu, \omega \in Val$ and $V, W \subseteq \mathcal{V}_{\mathrm{m}}$.*

Proof We prove that these valuations are equal by proving that they are equal for every $n \in \mathcal{V}_{\mathrm{m}}$ in their domain:

$$
\begin{aligned}
& m_W(m_W(\omega, \mu)\restriction_V, \omega\restriction_W)(n) \\
&= \begin{cases} m_W(\omega, \mu)\restriction_V (n) & \text{if } n \notin W \\ \omega\restriction_W (n) & \text{if } n \in W \end{cases} \\
&= \begin{cases} \omega\restriction_V (n) & \text{if } n \notin W \\ \mu\restriction_V (n) & \text{if } n \notin W \wedge n \in W \\ \omega\restriction_W (n) & \text{if } n \in W \end{cases} \\
&= \begin{cases} \omega\restriction_V (n) & \text{if } n \notin W \\ \omega\restriction_W (n) & \text{if } n \in W \end{cases}
\end{aligned}
$$

$$= \quad \omega{\restriction_{V \cup W}}\ (n)$$

$\boxtimes$

Lemma 39 *If* $\sigma = m_V(\sigma', \sigma'')$ *and* $\uparrow dom(\sigma) = \uparrow dom(\sigma') = \uparrow dom(\sigma'')$*, then* $\sigma(\uparrow) = m_V(\sigma'(\uparrow), \sigma''(\uparrow))$*, for any* $\sigma, \sigma', \sigma'' \in \Sigma$ *and* $V \subseteq \mathcal{V}_{\mathrm{m}}$*.*

Proof Suppose $\sigma = m_V(\sigma', \sigma'')$ and $\uparrow dom(\sigma) = \uparrow dom(\sigma') = \uparrow dom(\sigma'')$. Then, $\sigma(n)(t) = m_V(\sigma', \sigma'')(n)(t)$ for every $n \in \mathcal{V}_{\mathrm{m}}$ and every t in the domain of σ, in particular for $t = \uparrow dom(\sigma)$.

$$\begin{array}{rl} & \sigma(n)(\uparrow dom(\sigma)) \\ = & m_V(\sigma', \sigma'')(n)(\uparrow dom(\sigma)) \\ = & \begin{cases} \sigma'(n)(\uparrow dom(\sigma)) & \text{if } n \notin V \\ \sigma''(n)(\uparrow dom(\sigma)) & \text{if } n \in V \end{cases} \\ = & \begin{cases} \sigma'(n)(\uparrow dom(\sigma')) & \text{if } n \notin V \\ \sigma''(n)(\uparrow dom(\sigma'')) & \text{if } n \in V \end{cases} \\ = & m_V(\sigma'(n)(\uparrow dom(\sigma')), \sigma''(n)(\uparrow dom(\sigma''))) \end{array}$$

Since $\sigma(n)(\uparrow dom(\sigma)) = m_V(\sigma'(n)(\uparrow dom(\sigma')), \sigma''(n)(\uparrow dom(\sigma'')))$ for any $n \in \mathcal{V}_{\mathrm{m}}$, we conclude that $\sigma(\uparrow dom(\sigma)) = m_V(\sigma'(\uparrow dom(\sigma')), \sigma''(\uparrow dom(\sigma'')))$. $\boxtimes$

A.6 The Axiom: $[\![\ V\ \mid\ x\ ]\!] \approx x$ if $Var(x) \cap V = \emptyset$

Take $R \subseteq \mathcal{T} \times \mathcal{T}$ to be the relation

$$\begin{array}{rcl} R & = & \{(|[\ V : \nu \mid x\ ]|, x) \mid V \subseteq \mathcal{V}_{\mathrm{m}}, x \in \mathcal{T}, \nu \in Val, Var(x) \cap V = \emptyset\} \\ & \cup & \{([\![\ V\ \mid\ x\ ]\!], x) \mid V \subseteq \mathcal{V}_{\mathrm{m}}, x \in \mathcal{T}, Var(x) \cap V = \emptyset\} \\ & \cup & \{(x, x) \mid x \in \mathcal{T}\} \end{array}$$

For $(|[\ V : \nu \mid x\ ]|, x) \in R$ with $Var(x) \cap V = \emptyset$ we have the following cases:

1. $\langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle\,\checkmark$, which needs the hypothesis
   $\langle\, x, m_V(\mu, \nu)\,\rangle\,\checkmark$, so by lemma 42
   $\langle\, x, \mu\,\rangle\,\checkmark$.
2. $\langle\, x, \mu\,\rangle\,\checkmark$, so by lemma 42
   $\langle\, x, m_V(\mu, \nu)\,\rangle\,\checkmark$.
3. $\langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
   $\exists_{w,p',w'}\ \langle\, x, m_V(\mu, \nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p', w'\,\rangle$ with $l = m_V(w, \mu)$, $p = |[\ V : w'{\restriction_V}\ \mid p'\ ]|$, $\mu' = m_V(w', \mu)$, so by lemma 43
   $\nu(n) = w(n) = w'(n)$ for any $n \in V$, so
   $\langle\, x, m_V(\mu, \nu)\,\rangle \overset{a, m_V(w,\nu)}{\mapsto} \langle\, p', m_V(w', \nu)\,\rangle$, so by lemma 44
   $\langle\, x, m_V(\mu, \mu)\,\rangle \overset{a, m_V(w,\mu)}{\mapsto} \langle\, p', m_V(w', \mu)\,\rangle$, so by lemma 44
   $\langle\, x, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p', \mu'\,\rangle$.
   Note that $Var(p') \cap V = \emptyset$ by lemma 45, so $(|[\ V : w'{\restriction_V}\ \mid p'\ ]|, p') \in R$.
4. $\langle\, x, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu'\,\rangle$, so by lemma 43
   $\mu(n) = l(n) = \mu'(n)$ for any $n \in V$, so

$\langle\, x, m_V(\mu,\mu)\,\rangle \overset{a,m_V(l,\mu)}{\mapsto} \langle\, p, m_V(\mu',\mu)\,\rangle$, so by lemma 44
$\langle\, x, m_V(\mu,\nu)\,\rangle \overset{a,m_V(l,\nu)}{\mapsto} \langle\, p, m_V(\mu',\nu)\,\rangle$, so
$\langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{a,m_V(m_V(l,\nu),\mu)}{\mapsto} \langle\, |[\ V : m_V(\mu',\nu)\restriction_V \mid p\ ]|, m_V(m_V(\mu',\nu),\mu)\,\rangle$.

$$
\begin{aligned}
& m_V(m_V(l,\nu),\mu)(n) \\
= & \begin{cases} m_V(l,\nu)(n) & \text{if } n \notin V \\ \mu(n) & \text{if } n \in V \end{cases} \\
= & \begin{cases} l(n) & \text{if } n \notin V \\ \nu(n) & \text{if } n \notin V \wedge n \in V \\ \mu(n) & \text{if } n \in V \end{cases} \\
= & \begin{cases} l(n) & \text{if } n \notin V \\ \mu(n) & \text{if } n \in V \end{cases} \\
= & \begin{cases} l(n) & \text{if } n \notin V \\ l(n) & \text{if } n \in V \end{cases} \\
= & \ l(n)
\end{aligned}
$$

Similarly, $m_V(m_V(\mu',\nu),\mu) = \mu'$, so
$\langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, |[\ V : m_V(\mu',\nu)\restriction_V \mid p\ ]|, \mu'\,\rangle$.
Note that $Var(p) \cap V = \emptyset$ by lemma 45, so $(|[\ V : m_V(\mu',\nu)\restriction_V \mid p\ ]|, p) \in R$.

5. $\langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
$\exists_{\sigma,\sigma',p',w'}\ \langle\, x, m_V(\mu,\nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$ with $s = m_V(\sigma,\sigma')$, $p = |[\ V : w'\restriction_V \mid p'\ ]|$, $\mu' = m_V(w', \sigma'(\uparrow))$, so by lemma 47
$w' = \sigma(\uparrow)$, so by lemma 48
$\langle\, x, m_V(\mu,\mu)\,\rangle \overset{m_V(\sigma,\sigma')}{\rightsquigarrow} \langle\, p', m_V(\sigma(\uparrow), \sigma'(\uparrow))\,\rangle$, so
$\langle\, x, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p', \mu'\,\rangle$.
Note that $Var(p') \cap V = \emptyset$ by lemma 45, so $(|[\ V : w'\restriction_V \mid p'\ ]|, p') \in R$.
6. $\langle\, x, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, so by lemma 47
$\mu' = s(\uparrow)$, so
$\langle\, x, m_V(\mu,\mu)\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, s(\uparrow)\,\rangle$, so by lemma 48
$\langle\, x, m_V(\mu,\nu)\,\rangle \overset{m_V(s,s')}{\rightsquigarrow} \langle\, p', m_V(s(\uparrow), s'(\uparrow))\,\rangle$, so
$\langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{m_V(m_V(s,s'),s)}{\rightsquigarrow} \langle\, |[\ V : m_V(s(\uparrow), s'(\uparrow))\restriction_V \mid p'\ ]|, m_V(m_V(s(\uparrow), s'(\uparrow)), s(\uparrow))\,\rangle$
, so by lemma 41
$\langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V : m_V(s(\uparrow), s'(\uparrow))\restriction_V \mid p'\ ]|, s(\uparrow)\,\rangle$, so
$\langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V : m_V(s(\uparrow), s'(\uparrow))\restriction_V \mid p'\ ]|, \mu'\,\rangle$.
Note that $Var(p) \cap V = \emptyset$ by lemma 45, so $(|[\ V : m_V(s(\uparrow), s'(\uparrow))\restriction_V \mid p\ ]|, p) \in R$.

The proof for the case of $(|[\![\ V \mid x\ ]\!]|, x) \mid Var(x) \cap V = \emptyset) \in R$ is analogous to the case $(|[\ V : \nu \mid x\ ]|, x) \in R$ with $Var(x) \cap V = \emptyset$. For $(x, x) \in R$ the proof is trivial.

Lemma 40 *If $Var(d) \cap V = \emptyset$ and $(\mu, \mu') \models d$ then $(m_V(\mu,\nu), m_V(\mu',\nu)) \models d$, for any re-initialization clause d, $V \subseteq \mathcal{V}_{\mathrm{m}}$ and $\mu, \mu', \nu \in Val$.*

Proof Suppose $Var(d) \cap V = \emptyset$ and $(\mu, \mu') \models d$. Suppose d is of the form $[W \,|Pred\,]$. Then, by the definition of re-initialization clauses, $\mu(n) = \mu'(n)$ for all $n \notin W$ and $(\mu, \mu') \models Pred$. This

means that *Pred* evaluates to *true* for $n^- = \mu(n)$ and $n^+ = \mu'(n)$. Since no $n \in V$ occurs in *Pred*, *Pred* will still evaluate to *true* if we take an arbitrary ν and take $n^- = \nu(n)$ and $n^+ = \nu'(n)$ for any $n \in V$. Therefore, $(m_V(\mu,\nu), m_V(\mu',\nu)) \models \mathit{Pred}$ for any ν.

Furthermore, suppose $n \notin W$, then:

$$m_V(\mu,\nu)(n) \quad = \quad \begin{cases} \mu(n) & \text{if } n \notin V \\ \nu(n) & \text{if } n \in V \end{cases} \quad = \quad \begin{cases} \mu'(n) & \text{if } n \notin V \\ \nu(n) & \text{if } n \in V \end{cases} \quad = \quad m_V(\mu',\nu)(n)$$

$(m_V(\mu,\nu), m_V(\mu',\nu)) \models [W \,|\mathit{Pred}\,]$, so $(m_V(\mu,\nu), m_V(\mu',\nu)) \models d$. ⊠

Lemma 41 *$m_V(m_V(\mu,\nu),\mu) = \mu$ for any $V \subseteq \mathcal{V}_\mathrm{m}$ and $\mu,\nu \in \mathit{Val}$.*

Proof We prove that these valuations are equal by proving that they are equal for every $n \in \mathcal{V}_\mathrm{m}$ in their domain:

$$\begin{aligned} & m_V(m_V(\mu,\nu),\mu)(n) \\ = \quad & \begin{cases} m_V(\mu,\nu)(n) & \text{if } n \notin V \\ \mu(n) & \text{if } n \in V \end{cases} \\ = \quad & \begin{cases} \mu(n) & \text{if } n \notin V \\ \nu(n) & \text{if } n \notin V \wedge n \in V \\ \mu(n) & \text{if } n \in V \end{cases} \\ = \quad & \begin{cases} \mu(n) & \text{if } n \notin V \\ \mu(n) & \text{if } n \in V \end{cases} \\ = \quad & \mu(n) \end{aligned}$$

⊠

Lemma 42 *If $\mathit{Var}(x) \cap V = \emptyset$ then for any $x \in \mathcal{T}$, $V \subseteq \mathcal{V}_\mathrm{m}$ and $\mu,\nu \in \mathit{Val}$:*

$$\langle\, x,\mu \,\rangle\checkmark \iff \langle\, x, m_V(\mu,\nu) \,\rangle\checkmark$$

Proof We prove this by induction on the structure of the term x.
Suppose $\mathit{Var}(x) \cap V = \emptyset$ and $\langle\, x,\mu \,\rangle\checkmark$.

- δ, a, c: cannot terminate.
- ϵ: we can conclude immediately that $\langle\, \epsilon, m_V(\mu,\nu) \,\rangle\checkmark$.
- $d \gg p$: we need the hypothesis $\exists_{\mu'}(\mu,\mu') \models d, \langle\, p,\mu' \,\rangle\checkmark$.
  By lemma 40 we have that $(m_V(\mu,\nu), m_V(\mu',\nu)) \models d$ for any ν.
  Applying the induction hypothesis gives $\langle\, p, m_V(\mu',\nu') \,\rangle\checkmark$ for any ν'.
  We choose $\nu' = \nu$ and we conclude that $\langle\, d \gg p, m_V(\mu,\nu) \,\rangle\checkmark$.
- $p \oplus q$: we need one of the hypotheses
  - $\langle\, p,\mu \,\rangle\checkmark$.
    Applying the induction hypothesis gives $\langle\, p, m_V(\mu,\nu) \,\rangle\checkmark$.
    We conclude that $\langle\, p \oplus q, m_V(\mu,\nu) \,\rangle\checkmark$.

– $\langle\, q, \mu \,\rangle \checkmark$, which is similar to the previous case.

- $p \odot q$, $p \blacktriangleright q$, $p \rhd q$, $p \parallel q$, $p \between q$, $p \mid q$, $\partial_H(p)$: analogous to the case $p \oplus q$.

Suppose $Var(x) \cap V = \emptyset$ and $\langle\, x, m_V(\mu, \nu) \,\rangle \checkmark$.

- δ, a, c: cannot terminate.
- ϵ: we can conclude immediately that $\langle\, \epsilon, \mu \,\rangle \checkmark$.
- $d \gg p$: we need the hypothesis $\exists_{\mu'}(m_V(\mu, \nu), \mu') \models d, \langle\, p, \mu' \,\rangle \checkmark$.
  By lemma 40 we have that $(m_V(m_V(\mu, \nu), \nu'), m_V(\mu', \nu')) \models d$ for any ν'.
  We choose $\nu' = \mu$ and use lemma 41 which gives $(\mu, m_V(\mu', \mu)) \models d$.
  Applying the induction hypothesis gives $\langle\, p, m_V(\mu', \nu'') \,\rangle \checkmark$ for any ν''.
  We choose $\nu'' = \mu$, so $\langle\, p, m_V(\mu', \mu) \,\rangle \checkmark$.
  We conclude that $\langle\, d \gg p, \mu \,\rangle \checkmark$.
- $p \oplus q$: we need one of the hypotheses
  – $\langle\, p, m_V(\mu, \nu) \,\rangle \checkmark$.
    Applying the induction hypothesis gives $\langle\, p, \mu \,\rangle \checkmark$.
    We conclude that $\langle\, p \oplus q, \mu \,\rangle \checkmark$.
  – $\langle\, q, \mu \,\rangle \checkmark$, which is similar to the previous case.
- $p \odot q$, $p \blacktriangleright q$, $p \rhd q$, $p \parallel q$, $p \between q$, $p \mid q$, $\partial_H(p)$: analogous to the case $p \oplus q$.

$\boxtimes$

Lemma 43 *For any $x, x' \in \mathcal{T}$, $V \subseteq \mathcal{V}_{\mathrm{m}}$, $a \in \mathcal{A}$ and $\mu, \mu', l \in Val$:*

$$Var(x) \cap V = \emptyset, \langle\, x, \mu \,\rangle \stackrel{a,l}{\mapsto} \langle\, x', \mu' \,\rangle \implies \mu(n) = l(n) = \mu'(n) \text{ for any } n \in V.$$

Proof We prove this by induction on the structure of the term x. Suppose $Var(x) \cap V = \emptyset$ and $\langle\, x, \mu \,\rangle \stackrel{a,l}{\mapsto} \langle\, x', \mu' \,\rangle$.

- δ, ϵ, c: cannot perform an action transition.
- a: we need the hypothesis $\mu = l = \mu', x' = \epsilon$. Clearly, $\mu(n) = l(n) = \mu'(n)$ for any $n \in V$.
- $d \gg p$: we need the hypothesis $\exists_{\mu''}(\mu, \mu'') \models d, \langle\, p, \mu'' \,\rangle \stackrel{a,l}{\mapsto} \langle\, x', \mu' \,\rangle$.
  Applying the induction hypothesis gives $\mu''(n) = l(n) = \mu'(n)$ for any $n \in V$.
  Suppose $d = [W \,|Pred_d]$.
  Then, $\mu(n) = \mu''(n)$ for any $n \notin W$.
  $Var(x) \cap V = \emptyset \implies Var(d) \cap V = \emptyset \implies W \cap V = \emptyset$, so $n \in V \implies n \notin W$.
  Therefore, $\mu(n) = \mu''(n)$ for any $n \in V$.
  We conclude that $\mu(n) = l(n) = \mu'(n)$ for any $n \in V$.
- $p \oplus q$: we need one of the hypotheses
  – $\langle\, p, \mu \,\rangle \stackrel{a,l}{\mapsto} \langle\, x', \mu' \,\rangle$:
    Applying the induction hypothesis directly gives $\mu(n) = l(n) = \mu'(n)$ for any $n \in V$.
  – $\langle\, q, \mu \,\rangle \stackrel{a,l}{\mapsto} \langle\, x', \mu' \,\rangle$, which is similar to the previous case.

- $p \odot q$, $p \blacktriangleright q$, $p \rhd q$, $p \parallel q$, $p \between q$, $p \mid q$, $\partial_H(p)$: analogous to the case $p \oplus q$.

$\boxtimes$

Lemma 44 *If $Var(x) \cap V = \emptyset$ then for any $x, x' \in \mathcal{T}$, $V \subseteq \mathcal{V}_{\mathrm{m}}$, $a \in \mathcal{A}$ and $\mu, \mu', \nu, \nu', l \in Val$:*

$$\langle x, m_V(\mu,\nu)\rangle \overset{a,m_V(l,\nu)}{\mapsto} \langle x', m_V(\mu',\nu)\rangle \implies \langle x, m_V(\mu,\nu')\rangle \overset{a,m_V(l,\nu')}{\mapsto} \langle x', m_V(\mu',\nu')\rangle$$

Proof We prove this by induction on the structure of the term x. Suppose $Var(x) \cap V = \emptyset$ and $\langle x, m_V(\mu,\nu)\rangle \overset{a,m_V(l,\nu)}{\mapsto} \langle x', m_V(\mu',\nu)\rangle$.

- δ, ϵ, c: cannot perform an action transition.
- a: we need the hypothesis $m_V(\mu,\nu) = m_V(l,\nu) = m_V(\mu',\nu)$ and $x' = \epsilon$.
  This implies that $m_V(\mu,\nu') = m_V(l,\nu') = m_V(\mu',\nu')$ for any ν'.
  Since $\langle a, m_V(\mu,\nu')\rangle \overset{a,m_V(\mu,\nu')}{\mapsto} \langle \epsilon, m_V(\mu,\nu')\rangle$,
  we conclude that $\langle a, m_V(\mu,\nu')\rangle \overset{a,m_V(l,\nu')}{\mapsto} \langle x', m_V(\mu',\nu')\rangle$.
- $d \gg p$: we need the hypothesis $\exists_{\mu''}(m_V(\mu,\nu),\mu'') \models d$ and $\langle p, \mu''\rangle \overset{a,m_V(l,\nu)}{\mapsto} \langle x', m_V(\mu',\nu)\rangle$.
  By lemma 40, $(m_V(m_V(\mu,\nu),\nu'), m_V(\mu'',\nu')) \models d$ for any ν'.
  Therefore, $(m_V(\mu,\nu'), m_V(\mu'',\nu')) \models d$ for any ν'.
  In particular, $(m_V(\mu,\nu), m_V(\mu'',\nu)) \models d$, so there is a $\mu'' = m_V(\mu'',\nu)$.
  Therefore, $\langle p, m_V(\mu'',\nu)\rangle \overset{a,m_V(l,\nu)}{\mapsto} \langle x', m_V(\mu',\nu)\rangle$.
  Applying the induction hypothesis gives $\langle p, m_V(\mu'',\nu')\rangle \overset{a,m_V(l,\nu')}{\mapsto} \langle x', m_V(\mu',\nu')\rangle$.
  We conclude that $\langle d \gg p, m_V(\mu'',\nu')\rangle \overset{a,m_V(l,\nu')}{\mapsto} \langle x', m_V(\mu',\nu')\rangle$.
- $p \oplus q$: we need one of the hypotheses
  - $\langle p, m_V(\mu,\nu)\rangle \overset{a,m_V(l,\nu)}{\mapsto} \langle x', m_V(\mu',\nu)\rangle$.
    Applying the induction hypothesis gives $\langle p, m_V(\mu,\nu')\rangle \overset{a,m_V(l,\nu')}{\mapsto} \langle x', m_V(\mu',\nu')\rangle$.
    We conclude that $\langle p \oplus q, m_V(\mu,\nu')\rangle \overset{a,m_V(l,\nu')}{\mapsto} \langle x', m_V(\mu',\nu')\rangle$.
  - $\langle q, m_V(\mu,\nu)\rangle \overset{a,m_V(l,\nu)}{\mapsto} \langle x', m_V(\mu',\nu)\rangle$, which is similar to the previous case.
- $p \odot q$, $p \blacktriangleright q$, $p \rhd q$, $p \parallel q$, $p \between q$, $p \mid q$, $\partial_H(p)$: analogous to the case $p \oplus q$.

$\boxtimes$

Lemma 45 *If $Var(x) \cap V = \emptyset$ and $\langle x, \mu\rangle \xrightarrow{l} \langle x', \mu'\rangle$, then $Var(x') \cap V = \emptyset$.*

Proof Suppose $Var(x) \cap V = \emptyset$ and $\langle x, \mu\rangle \xrightarrow{l} \langle x', \mu'\rangle$. By checking the HyPA and abstraction semantics, it can be verified that $Var(x') \subseteq Var(x)$. Therefore, $Var(x') \cap V = \emptyset$. $\boxtimes$

Lemma 46 *For any flow clause c, $V \subseteq \mathcal{V}_{\mathrm{m}}$, $\sigma, \sigma' \in \Sigma$ and $\mu, \nu, \nu' \in Val$:*

$$Var(c) \cap V = \emptyset, (\mu,\sigma) \models c \implies (m_V(\mu,\nu'), m_V(\sigma,\sigma')) \models c$$

Proof Suppose $Var(c) \cap V = \emptyset$ and $(\mu, \sigma) \models c$. Suppose c is of the form $\left(\begin{array}{c|c} W & Pred \end{array} \right)$, then $\sigma \models Pred$ and $\mu(n) = \sigma(0)(n)$ for all $n \in W$ by the definition of flow clauses.

Since no $n \in V$ occurs in *Pred*, *Pred* will have exactly the same solutions if we take the flow of a variable $n \in V$ from an arbitrary flow σ'. Therefore, $m_V(\sigma, \sigma') \models Pred$ for any σ'.

Furthermore, suppose $n \in W$. Then $n \in Var(c)$, so we know that $n \notin V$. For $n \notin V$ and arbitrary ν':

$$\begin{aligned} m_V(\mu, \nu')(n) &= \begin{cases} \mu(n) & \text{if } n \notin V \\ \nu'(n) & \text{if } n \in V \end{cases} \\ &= \mu(n) \\ &= \sigma(0)(n) \\ &= \begin{cases} \sigma(0)(n) & \text{if } n \notin V \\ \sigma'(0)(n) & \text{if } n \in V \end{cases} \\ &= m_V(\sigma, \sigma')(0)(n) \end{aligned}$$

$(m_V(\mu, \nu'), m_V(\sigma, \sigma')) \models \left(\begin{array}{c|c} W & Pred \end{array} \right)$, so $(m_V(\mu, \nu'), m_V(\sigma, \sigma')) \models c$. ⊠

Lemma 47 *For any $x, x' \in \mathcal{T}$, $V \subseteq \mathcal{V}_\mathrm{m}$, $\sigma \in \Sigma$ and $\mu, \nu' \in Val$:*

$$\langle\, x, \mu \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, x', \mu' \,\rangle \implies \mu' = \sigma(\uparrow)$$

Proof This can be verified by checking the HyPA and abstraction semantics. ⊠

Lemma 48 *If $Var(x) \cap V = \emptyset$ then for any $x, x' \in \mathcal{T}$, $V \subseteq \mathcal{V}_\mathrm{m}$, $\sigma, \sigma' \in \Sigma$ and $\mu, \nu, \nu' \in Val$:*

$$\langle\, x, m_V(\mu, \nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, x', \sigma(\uparrow) \,\rangle \implies \langle\, x, m_V(\mu, \nu') \,\rangle \overset{m_V(\sigma,\sigma')}{\rightsquigarrow} \langle\, x', m_V(\sigma(\uparrow), \sigma'(\uparrow)) \,\rangle$$

Proof We prove this by induction on the structure of the term x. Suppose $Var(x) \cap V = \emptyset$ and $\langle\, x, m_V(\mu, \nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, x', \sigma(\uparrow) \,\rangle$.

- δ, ϵ, a: cannot perform a flow transition.
- c: we need the hypothesis $(m_V(\mu, \nu), \sigma) \models c$ and $x' = c$.
  By lemma 46, $(m_V(m_V(\mu, \nu), \nu'), m_V(\sigma, \sigma')) \models c$ for any ν' and σ'.
  Therefore, $(m_V(\mu, \nu'), m_V(\sigma, \sigma')) \models c$.
  We conclude that $\langle\, c, m_V(\mu, \nu') \,\rangle \overset{m_V(\sigma,\sigma')}{\rightsquigarrow} \langle\, x', m_V(\sigma(\uparrow), \sigma'(\uparrow)) \,\rangle$.
- $d \gg p$: we need the hypothesis $\exists_{\mu'}(m_V(\mu, \nu), \mu') \models d$ and $\langle\, p, \mu' \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, x', \sigma(\uparrow) \,\rangle$.
  By lemma 40, $(m_V(m_V(\mu, \nu), \nu'), m_V(\mu', \nu')) \models d$ for any ν'.
  Therefore, $(m_V(\mu, \nu'), m_V(\mu', \nu')) \models d$ for any ν'.
  In particular, $(m_V(\mu, \nu), m_V(\mu', \nu)) \models d$, so there is a $\mu' = m_V(\mu', \nu)$.
  Therefore, $\langle\, p, m_V(\mu', \nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, x', \sigma(\uparrow) \,\rangle$.
  Applying the induction hypothesis gives $\langle\, p, m_V(\mu', \nu') \,\rangle \overset{m_V(\sigma,\sigma')}{\rightsquigarrow} \langle\, x', m_V(\sigma(\uparrow), \sigma'(\uparrow)) \,\rangle$ for any ν' and σ'.
  We conclude that $\langle\, d \gg p, m_V(\mu', \nu') \,\rangle \overset{m_V(\sigma,\sigma')}{\rightsquigarrow} \langle\, x', m_V(\sigma(\uparrow), \sigma'(\uparrow)) \,\rangle$.

- $p \oplus q$: we need one of the hypotheses
  - $\langle\, p, m_V(\mu,\nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, x', \sigma(\uparrow)\,\rangle$.
    Applying the induction hypothesis gives $\langle\, p, m_V(\mu,\nu')\,\rangle \overset{m_V(\sigma,\sigma')}{\rightsquigarrow} \langle\, x', m_V(\sigma(\uparrow),\sigma'(\uparrow))\,\rangle$.
    We conclude that $\langle\, p \oplus q, m_V(\mu,\nu')\,\rangle \overset{m_V(\sigma,\sigma')}{\rightsquigarrow} \langle\, x', m_V(\sigma(\uparrow),\sigma'(\uparrow))\,\rangle$.
  - $\langle\, q, m_V(\mu,\nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, x', \sigma(\uparrow)\,\rangle$, which is similar to the previous case.
- $p \odot q$, $p \blacktriangleright q$, $p \rhd q$, $p \parallel q$, $p \lfloor\!\lfloor q$, $p \mid q$, $\partial_H(p)$: analogous to the case $p \oplus q$.

$\boxtimes$

A.7 The Axiom: $[\![\ V\ \mid\ x\ ]\!] \parallel y \approx [\![\ V\ \mid\ x \parallel y\ ]\!]$ if $Var(y) \cap V = \emptyset$

Take $R \subseteq \mathcal{T} \times \mathcal{T}$ to be the relation

$$\begin{array}{rcl} R & = & \{(|[\ V : \nu \mid x\ ]| \parallel y, |[\ V : \nu \mid x \parallel y\ ]|) \mid V \subseteq \mathcal{V}_{\mathrm{m}}, x, y \in \mathcal{T}, \nu \in Val,\ Var(y) \cap V = \emptyset\} \\ & \cup & \{(y, |[\ V : \nu \mid y\ ]|) \mid V \subseteq \mathcal{V}_{\mathrm{m}}, y \in \mathcal{T}, \nu \in Val,\ Var(y) \cap V = \emptyset\} \\ & \cup & \{([\![\ V\ \mid\ x\ ]\!] \parallel y, [\![\ V\ \mid\ x \parallel y\ ]\!]) \mid V \subseteq \mathcal{V}_{\mathrm{m}}, x, y \in \mathcal{T},\ Var(y) \cap V = \emptyset\} \\ & \cup & \{(x, x) \mid x \in \mathcal{T}\} \end{array}$$

For $(|[\ V : \nu \mid x\ ]| \parallel y, |[\ V : \nu \mid x \parallel y\ ]|) \mid Var(y) \cap V = \emptyset) \in R$ we have the following cases:

1. $\langle\, |[\ V : \nu \mid x\ ]| \parallel y, \mu\,\rangle\checkmark$, which needs the hypothesis
   $\langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle\checkmark$ and $\langle\, y, \mu\,\rangle\checkmark$, which needs the hypothesis
   $\langle\, x, m_V(\mu,\nu)\,\rangle\checkmark$.
   $\langle\, y, m_V(\mu,\nu)\,\rangle\checkmark$ by lemma 42 , so
   $\langle\, x \parallel y, m_V(\mu,\nu)\,\rangle\checkmark$, so
   $\langle\, |[\ V : \nu \mid x \parallel y\ ]|, \mu\,\rangle\checkmark$.
2. $\langle\, |[\ V : \nu \mid x \parallel y\ ]|, \mu\,\rangle\checkmark$, which needs the hypothesis
   $\langle\, x \parallel y, m_V(\mu,\nu)\,\rangle\checkmark$, which needs the hypothesis
   $\langle\, x, m_V(\mu,\nu)\,\rangle\checkmark$ and $\langle\, y, m_V(\mu,\nu)\,\rangle\checkmark$, so by lemma 42
   $\langle\, x, m_V(\mu,\nu)\,\rangle\checkmark$ and $\langle\, y, \mu\,\rangle\checkmark$, so
   $\langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle\checkmark$ and $\langle\, y, \mu\,\rangle\checkmark$, so
   $\langle\, |[\ V : \nu \mid x\ ]| \parallel y, \mu\,\rangle\checkmark$.
3. $\langle\, |[\ V : \nu \mid x\ ]| \parallel y, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu'\,\rangle$, which needs one of the hypotheses
   (a) $\exists_{p'}\ \langle\, |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{a,l}{\mapsto}$
   $\langle\, p', \mu'\,\rangle$ with $p = p' \parallel y$, which needs the hypothesis
   $\exists_{w,p'',w'}\ \langle\, x, m_V(\mu,\nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p'', w'\,\rangle$ with $l = m_V(w,\mu)$, $p' = |[\ V : w'\restriction_V \mid p''\ ]|$, $\mu' = m_V(w',\mu)$, so
   $\langle\, x \parallel y, m_V(\mu,\nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p'' \parallel y, w'\,\rangle$, so
   $\langle\, |[\ V : \nu \mid x \parallel y\ ]|, \mu\,\rangle \overset{a,m_V(w,\mu)}{\mapsto} \langle\, |[\ V : w'\restriction_V \mid p'' \parallel y\ ]|, m_V(w',\mu)\,\rangle$, so
   $\langle\, |[\ V : \nu \mid x \parallel y\ ]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, |[\ V : w'\restriction_V \mid p'' \parallel y\ ]|, \mu'\,\rangle$.
   Recall that $p = p' \parallel y = |[\ V : w'\restriction_V \mid p''\ ]| \parallel y$ and note that $(|[\ V : w'\restriction_V \mid p''\ ]| \parallel y, |[\ V : w'\restriction_V \mid p'' \parallel y\ ]|) \in R$
   (b) $\exists_{p'}\ \langle\, y, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p', \mu'\,\rangle$ with $p = |[\ V : \nu \mid x\ ]| \parallel p'$, so by lemma 43
   $\mu(n) = l(n) = \mu'(n)$ for any $n \in V$, so
   $\langle\, y, m_V(\mu,\mu)\,\rangle \overset{a,m_V(l,\mu)}{\mapsto} \langle\, p', m_V(\mu',\mu)\,\rangle$, so by lemma 44
   $\langle\, y, m_V(\mu,\nu)\,\rangle \overset{a,m_V(w,\nu)}{\mapsto} \langle\, p', m_V(w',\nu)\,\rangle$, so

$\langle\, x \parallel y, m_V(\mu,\nu)\,\rangle \overset{a,m_V(w,\nu)}{\mapsto} \langle\, x \parallel p', m_V(w',\nu)\,\rangle$, so

$\langle\, |[\, V : \nu \mid x \parallel y\,]|, \mu\,\rangle \overset{a,m_V(m_V(w,\nu),\mu)}{\mapsto} \langle\, |[\, V : m_V(w',\nu)\restriction_V \mid x \parallel p'\,]|, m_V(m_V(w',\nu),\mu)\,\rangle$, so

$\langle\, |[\, V : \nu \mid x \parallel y\,]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, |[\, V : \nu \mid x \parallel p'\,]|, \mu'\,\rangle$.

Note that by lemma 45 $Var(p') \cap V = \emptyset$, so $(|[\, V : \nu \mid x\,]| \parallel p', |[\, V : \nu \mid x \parallel p'\,]|) \in R$.

(c) $\exists_{p',q'}\ \langle\, |[\, V : \nu \mid x\,]|, \mu\,\rangle \overset{a',l}{\mapsto} \langle\, p', \mu'\,\rangle,\ \langle\, y, \mu\,\rangle \overset{a'',l}{\mapsto} \langle\, q', \mu'\,\rangle$ and $a = a' \gamma a''$ with $p = p' \parallel q'$, so

$\exists_{w,p'',w'}\ \langle\, x, m_V(\mu,\nu)\,\rangle \overset{a',w}{\mapsto} \langle\, p'', w'\,\rangle$ with $l = m_V(w,\mu),\ p' = |[\, V : w'\restriction_V \mid p''\,]|,\ \mu' = m_V(w',\mu)$, so by lemma 43

$\nu(n) = w(n) = w'(n)$ and $\mu(n) = l(n) = \mu'(n)$ for any $n \in V$, so

$\langle\, y, m_V(\mu,\mu)\,\rangle \overset{a'',m_V(l,\mu)}{\mapsto} \langle\, q', m_V(\mu',\mu)\,\rangle$, so by lemma 44

$\langle\, y, m_V(\mu,\nu)\,\rangle \overset{a'',m_V(l,\nu)}{\mapsto} \langle\, q', m_V(\mu',\nu)\,\rangle$.

$m_V(l,\nu) = m_V(m_V(w,\mu),\nu) = m_V(w,\nu) = m_V(w,w) = w$ and similarly, $m_V(\mu',\nu) = w'$, so

$\langle\, y, m_V(\mu,\nu)\,\rangle \overset{a'',w}{\mapsto} \langle\, q', w'\,\rangle$, so

$\langle\, x \parallel y, m_V(\mu,\nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p'' \parallel q', w'\,\rangle$, so

$\langle\, |[\, V : \nu \mid x \parallel y\,]|, \mu\,\rangle \overset{a,m_V(w,\mu)}{\mapsto} \langle\, |[\, V : w'\restriction_V \mid p'' \parallel q'\,]|, m_V(w',\mu)\,\rangle$, so

$\langle\, |[\, V : \nu \mid x \parallel y\,]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, |[\, V : w'\restriction_V \mid p'' \parallel q'\,]|, \mu'\,\rangle$.

Recall that $p = p' \parallel q' = |[\, V : w'\restriction_V \mid p''\,]| \parallel q'$ and note that, by lemma 45 $Var(q') \cap V = \emptyset$, so $(|[\, V : w'\restriction_V \mid p''\,]| \parallel q', |[\, V : w'\restriction_V \mid p'' \parallel q'\,]|) \in R$

4. $\langle\, |[\, V : \nu \mid x \parallel y\,]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
$\exists_{w,p',w'}\ \langle\, x \parallel y, m_V(\mu,\nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p', w'\,\rangle$ with $l = m_V(w,\mu),\ p = |[\, V : w'\restriction_V \mid p'\,]|,\ \mu' = m_V(w',\mu)$, which needs one of the hypotheses

(a) $\exists_{p''}\ \langle\, x, m_V(\mu,\nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p'', w'\,\rangle$ with $p' = p'' \parallel y$, so

$\langle\, |[\, V : \nu \mid x\,]|, \mu)\,\rangle \overset{a,m_V(w,\mu)}{\mapsto} \langle\, |[\, V : w'\restriction_V \mid p''\,]|, m_V(w',\mu)\,\rangle$, so

$\langle\, |[\, V : \nu \mid x\,]|, \mu)\,\rangle \overset{a,l}{\mapsto} \langle\, |[\, V : w'\restriction_V \mid p''\,]|, \mu'\,\rangle$, so

$\langle\, |[\, V : \nu \mid x\,]| \parallel y, \mu)\,\rangle \overset{a,l}{\mapsto} \langle\, |[\, V : w'\restriction_V \mid p''\,]| \parallel y, \mu'\,\rangle$.

Recall that $p = |[\, V : w'\restriction_V \mid p'\,]| = |[\, V : w'\restriction_V \mid p'' \parallel y\,]|$ and note that $(|[\, V : w'\restriction_V \mid p''\,]| \parallel y, |[\, V : w'\restriction_V | \ p'' \parallel y\,]|) \in R$.

(b) $\exists_{p''}\ \langle\, y, m_V(\mu,\nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p'', w'\,\rangle$ with $p' = x \parallel p''$, so by lemma 43

$\nu(n) = w(n) = w'(n)$ for any $n \in V$, so

$\langle\, y, m_V(\mu,\nu)\,\rangle \overset{a,m_V(w,\nu)}{\mapsto} \langle\, p'', m_V(w',\nu)\,\rangle$, so by lemma 44

$\langle\, y, m_V(\mu,\mu)\,\rangle \overset{a,m_V(w,\mu)}{\mapsto} \langle\, p'', m_V(w',\mu)\,\rangle$, so

$\langle\, y, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p'', \mu'\,\rangle$, so

$\langle\, |[\, V : \nu \mid x\,]| \parallel y, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, |[\, V : \nu \mid x\,]| \parallel p'', \mu'\,\rangle$.

Recall that $p = |[\, V : w'\restriction_V \mid p'\,]| = |[\, V : w'\restriction_V \mid x \parallel p''\,]|$ and note that by lemma 45 $Var(p'') \cap V = \emptyset$, so $(|[\, V : w'\restriction_V \mid x\,]| \parallel p'', |[\, V : w'\restriction_V \mid x \parallel p''\,]|) \in R$

(c) $\exists_{p'',q''}\ \langle\, x, m_V(\mu,\nu)\,\rangle \overset{a',w}{\mapsto} \langle\, p'', w'\,\rangle,\ \langle\, y, m_V(\mu,\nu)\,\rangle \overset{a'',w}{\mapsto} \langle\, q'', w'\,\rangle$ and
$a = a' \gamma a''$ with $p' = p'' \parallel q''$, so by lemma 43

$\nu(n) = w(n) = w'(n)$ for any $n \in V$, so

$\langle\, y, m_V(\mu,\nu)\,\rangle \overset{a'',m_V(w,\nu)}{\mapsto} \langle\, q'', m_V(w',\nu)\,\rangle$, so by lemma 44

$\langle\, y, m_V(\mu,\mu)\,\rangle \overset{a'',m_V(w,\mu)}{\mapsto} \langle\, q'', m_V(w',\mu)\,\rangle$, so

$\langle\, y, \mu\,\rangle \overset{a'',l}{\mapsto} \langle\, q'', \mu'\,\rangle$.

$\langle\, |[\, V : \nu \mid x\,]|, \mu)\,\rangle \overset{a',m_V(w,\mu)}{\mapsto} \langle\, |[\, V : w'\restriction_V \mid p''\,]|, m_V(w',\mu)\,\rangle$, so

$\langle\, |[\, V : \nu \mid x\,]|, \mu)\,\rangle \overset{a',l}{\mapsto} \langle\, |[\, V : w'\restriction_V \mid p''\,]|, \mu'\,\rangle$, so

$\langle\, |[\ V : \nu \mid x\ ]| \parallel y, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, |[\ V : w'\restriction_V \mid p''\ ]| \parallel q'', \mu' \,\rangle$.
Recall that $p = |[\ V : w'\restriction_V \mid p'\ ]| = |[\ V : w'\restriction_V \mid p'' \parallel q''\ ]|$ and note that by lemma 45 $Var(q'') \cap V = \emptyset$, so $(|[\ V : w'\restriction_V \mid p''\ ]| \parallel q'', |[\ V : w'\restriction_V \mid p'' \parallel q''\ ]|) \in R$

5. $\langle\, |[\ V : \nu \mid x\ ]| \parallel y, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$, which needs one of the hypotheses

   (a) $\langle\, |[\ V : \nu \mid x\ ]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$ and $\langle\, y, \mu \,\rangle \checkmark$, which needs the hypothesis
   $\exists_{\sigma,\sigma',p',w'} \langle\, x, m_V(\mu, \nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w' \,\rangle$ with $s = m_V(\sigma, \sigma')$, $p = |[\ V : w'\restriction_V \mid p'\ ]|$, $\mu' = m_V(w', \sigma'(\uparrow))$, so by lemma 42
   $\langle\, y, m_V(\mu, \nu) \,\rangle \checkmark$, so
   $\langle\, x \parallel y, m_V(\mu, \nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w' \,\rangle$, so
   $\forall_{\sigma''} \langle\, |[\ V : \nu \mid x \parallel y\ ]|, \mu \,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p'\ ]|, m_V(w', \sigma''(\uparrow)) \,\rangle$.
   We take $\sigma'' = \sigma'$, so
   $\langle\, |[\ V : \nu \mid x \parallel y\ ]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p'\ ]|, \mu' \,\rangle$.
   Note that $(|[\ V : w'\restriction_V \mid p'\ ]|, |[\ V : w'\restriction_V \mid p'\ ]|) \in R$.

   (b) $\langle\, y, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$ and $\langle\, |[\ V : \nu \mid x\ ]|, \mu \,\rangle \checkmark$, which needs the hypothesis
   $\langle\, x, m_V(\mu, \nu) \,\rangle \checkmark$.
   $\mu' = s(\uparrow)$ by lemma 47 , so
   $\langle\, y, m_V(\mu, \mu) \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, s(\uparrow) \,\rangle$, so by lemma 48
   $\langle\, y, m_V(\mu, \nu) \,\rangle \overset{m_V(s,\sigma'')}{\rightsquigarrow} \langle\, p, m_V(s(\uparrow), \sigma''(\uparrow)) \,\rangle$ for any σ'' , so
   $\langle\, x \parallel y, m_V(\mu, \nu) \,\rangle \overset{m_V(s,\sigma'')}{\rightsquigarrow} \langle\, p, m_V(s(\uparrow), \sigma''(\uparrow)) \,\rangle$, so
   $\forall_{\sigma'''} \langle\, |[\ V : \nu \mid x \parallel y\ ]|, \mu \,\rangle \overset{m_V(m_V(s,\sigma''),\sigma''')}{\rightsquigarrow} \langle\, |[\ V : m_V(s(\uparrow), \sigma''(\uparrow))\restriction_V \mid p\ ]|, m_V(m_V(s(\uparrow), \sigma''(\uparrow)), \sigma'''(\uparrow)) \,\rangle$.
   We take $\sigma''' = \sigma'$, so
   $\langle\, |[\ V : \nu \mid x \parallel y\ ]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V : m_V(s(\uparrow), \sigma''(\uparrow))\restriction_V \mid p\ ]|, \mu' \,\rangle$.
   Note that by lemma 45 $Var(p) \cap V = \emptyset$, so $(|[\ V : m_V(s(\uparrow), \sigma''(\uparrow))\restriction_V \mid p\ ]|, p) \in R$.

   (c) $\exists_{p',q'} \langle\, |[\ V : \nu \mid x\ ]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p', \mu' \,\rangle$ and $\langle\, y, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, q', \mu' \,\rangle$ with $p = p' \parallel q'$ which needs the hypothesis $\exists_{\sigma,\sigma',p'',w'} \langle\, x, m_V(\mu, \nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'', w' \,\rangle$ with $s = m_V(\sigma, \sigma')$, $p' = |[\ V : w'\restriction_V \mid p''\ ]|$, $\mu' = m_V(w', \sigma'(\uparrow))$.
   $\mu' = s(\uparrow)$ and $w' = \sigma(\uparrow)$ by lemma 47 , so
   $\langle\, y, m_V(\mu, \mu) \,\rangle \overset{s}{\rightsquigarrow} \langle\, q', s(\uparrow) \,\rangle$, so by lemma 48
   $\langle\, y, m_V(\mu, \nu) \,\rangle \overset{m_V(s,\sigma'')}{\rightsquigarrow} \langle\, q', m_V(s(\uparrow), \sigma''(\uparrow)) \,\rangle$ for any σ''.
   We take $\sigma'' = \sigma$, so
   $m_V(s, \sigma'') = m_V(s, \sigma) = m_V(m_V(\sigma, \sigma'), \sigma) = m_V(\sigma, \sigma) = \sigma$ and similarly, $m_V(s(\uparrow), \sigma''(\uparrow)) = \sigma(\uparrow) = w'$, so
   $\langle\, y, m_V(\mu, \nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, q', w' \,\rangle$, so
   $\langle\, x \parallel y, m_V(\mu, \nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'' \parallel q', w' \,\rangle$, so
   $\forall_{\sigma'''} \langle\, |[\ V : \nu \mid x \parallel y\ ]|, \mu \,\rangle \overset{m_V(\sigma,\sigma''')}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p'' \parallel q'\ ]|, m_V(w', \sigma'''(\uparrow)) \,\rangle$.
   We take $\sigma''' = \sigma'$, so
   $\langle\, |[\ V : \nu \mid x \parallel y\ ]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p'' \parallel q'\ ]|, \mu' \,\rangle$.
   Recall that $p = p' \parallel q' = |[\ V : w'\restriction_V \mid p''\ ]| \parallel q'$ and note that by lemma 45 $Var(q') \cap V = \emptyset$, so $(|[\ V : w'\restriction_V \mid p''\ ]| \parallel q', |[\ V : w'\restriction_V \mid p'' \parallel q'\ ]|) \in R$

6. $\langle\, |[\ V : \nu \mid x \parallel y\ ]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
   $\exists_{\sigma,\sigma',p',w'} \langle\, x \parallel y, m_V(\mu, \nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w' \,\rangle$ with $s = m_V(\sigma, \sigma')$, $p = |[\ V : w'\restriction_V \mid p'\ ]|$, $\mu' = m_V(w', \sigma'(\uparrow))$, which needs one of the hypotheses

   (a) $\langle\, x, m_V(\mu, \nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w' \,\rangle$ and $\langle\, y, m_V(\mu, \nu) \,\rangle \checkmark$, so
   $\forall_{\sigma''} \langle\, |[\ V : \nu \mid x\ ]|, \mu \,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p'\ ]|, m_V(w', \sigma''(\uparrow)) \,\rangle$.
   We take $\sigma'' = \sigma'$, so
   $\langle\, |[\ V : \nu \mid x\ ]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p'\ ]|, \mu' \,\rangle$.
   $\langle\, y, \mu \,\rangle \checkmark$ by lemma 42 , so

$\langle\, |[\, V : \nu \mid x \,]| \parallel y, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\, V : w'\restriction_V \mid p' \,]|, \mu' \,\rangle$.
Note that by lemma 45 $Var(p') \cap V = \emptyset$, so $(|[\, V : w'\restriction_V \mid p' \,]|, |[\, V : w'\restriction_V \mid p' \,]|) \in R$.

(b) $\langle\, y, m_V(\mu,\nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w' \,\rangle$ and $\langle\, x, m_V(\mu,\nu) \,\rangle \checkmark$, so
$\langle\, |[\, V : \nu \mid x \,]|, \mu \,\rangle \checkmark$.
$w' = \sigma(\uparrow)$ by lemma 47 , so
$\langle\, y, m_V(\mu,\nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', \sigma(\uparrow) \,\rangle$, so by lemma 48
$\langle\, y, m_V(\mu,\mu) \,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, p', m_V(\sigma(\uparrow), \sigma''(\uparrow)) \,\rangle$ for any σ''.
We take $\sigma'' = \sigma'$, so
$\langle\, y, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p', \mu' \,\rangle$, so
$\langle\, |[\, V : \nu \mid x \,]| \parallel y, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, p', \mu' \,\rangle$.
Note that by lemma 45 $Var(p') \cap V = \emptyset$, so $(|[\, V : w'\restriction_V \mid p' \,]|, p') \in R$.

(c) $\exists_{p'',q'} \langle\, x, m_V(\mu,\nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'', w' \,\rangle$ and $\langle\, y, m_V(\mu,\nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, q', w' \,\rangle$ with $p' = p'' \parallel q'$, so
$\forall_{\sigma''} \langle\, |[\, V : \nu \mid x \,]|, \mu \,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\, V : w'\restriction_V \mid p'' \,]|, m_V(w', \sigma''(\uparrow)) \,\rangle$.
We take $\sigma'' = \sigma'$, so
$\langle\, |[\, V : \nu \mid x \,]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\, V : w'\restriction_V \mid p'' \,]|, \mu' \,\rangle$.
$w' = \sigma(\uparrow)$ by lemma 47 , so
$\langle\, y, m_V(\mu,\nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, q', \sigma(\uparrow) \,\rangle$, so by lemma 48
$\langle\, y, m_V(\mu,\mu) \,\rangle \overset{m_V(\sigma,\sigma''')}{\rightsquigarrow} \langle\, q', m_V(w', \sigma''') \,\rangle$ for any σ'''.
We take $\sigma''' = \sigma'$, so
$\langle\, y, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, q', \mu' \,\rangle$, so
$\langle\, |[\, V : \nu \mid x \,]| \parallel y, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\, V : w'\restriction_V \mid p'' \,]| \parallel q', \mu' \,\rangle$.
Recall that $p = |[\, V : w'\restriction_V \mid p' \,]| = |[\, V : w'\restriction_V \mid p'' \parallel q' \,]|$ and note that by lemma 45 $Var(q') \cap V = \emptyset$, so $(|[\, V : w'\restriction_V \mid p'' \,]| \parallel q', |[\, V : w'\restriction_V \mid p'' \parallel q' \,]|) \in R$

The proof for the case of $(y, |[\, V : \nu \mid y \,]|) \in R$ with $Var(y) \cap V = \emptyset$ is similar to the proof of the axiom $[\![\, V \mid x \,]\!] \approx x$ if $Var(x) \cap V = \emptyset$. The proof for the case of $([\![\, V \mid x \,]\!] \parallel y, [\![\, V \mid x \parallel y \,]\!]) \in R$ with $Var(y) \cap V = \emptyset$ is analogous to the case $(|[\, V : \nu \mid x \,]| \parallel y, |[\, V : \nu \mid x \parallel y \,]|) \in R$ with $Var(y) \cap V = \emptyset$. For $(x, x) \in R$ the proof is trivial.

A.8 The Axiom: $d \gg [\![\, V \mid x \,]\!] \approx [\![\, V \mid d \gg x \,]\!]$ if $Var(d) \cap V = \emptyset$

Take $R \subseteq \mathcal{T} \times \mathcal{T}$ to be the relation

$$
\begin{array}{rcl}
R & = & \{(d \gg |[\, V : \nu \mid x \,]|, |[\, V : \nu \mid d \gg x \,]|) \\
 & & \mid V \subseteq \mathcal{V}_{\mathrm{m}}, x \in \mathcal{T}, \nu \in \mathit{Val},\ \text{re-init. clause } d,\ \mathit{Var}(d) \cap V = \emptyset\} \\
 & \cup & \{(d \gg [\![\, V \mid x \,]\!], [\![\, V \mid d \gg x \,]\!]) \mid V \subseteq \mathcal{V}_{\mathrm{m}}, x \in \mathcal{T},\ \text{re-init. clause } d,\ \mathit{Var}(d) \cap V = \emptyset\} \\
 & \cup & \{(x, x) \mid x \in \mathcal{T}\}
\end{array}
$$

For $(d \gg |[\, V : \nu \mid x \,]|, |[\, V : \nu \mid d \gg x \,]|)$ with $Var(d) \cap V = \emptyset$ we ahve the following cases:

1. $\langle\, d \gg |[\, V : \nu \mid x \,]|, \mu \,\rangle \checkmark$, which needs the hypothesis
   $\exists_{\mu'}\ (\mu, \mu') \models d$ and $\langle\, |[\, V : \nu \mid x \,]|, \mu' \,\rangle \checkmark$, which needs the hypothesis
   $\langle\, x, m_V(\mu', \nu) \,\rangle \checkmark$.
   $(m_V(\mu,\nu), m_V(\mu',\nu)) \models d$ by lemma 40 , so
   $\langle\, d \gg x, m_V(\mu,\nu) \,\rangle \checkmark$, so
   $\langle\, |[\, V : \nu \mid d \gg x \,]|, \mu \,\rangle \checkmark$.
2. $\langle\, |[\, V : \nu \mid d \gg x \,]|, \mu \,\rangle \checkmark$, which needs the hypothesis
   $\langle\, d \gg x, m_V(\mu,\nu) \,\rangle \checkmark$, which needs the hypothesis
   $\exists_{\mu'}\ (m_V(\mu,\nu), \mu') \models d$ and $\langle\, x, \mu' \,\rangle \checkmark$.
   $\mu'\restriction_V = \nu$ by lemma 49 , so
   $\langle\, x, m_V(\mu', \nu) \,\rangle \checkmark$, so

$\langle\, x, m_V(m_V(\mu', \mu), \nu)\,\rangle \checkmark$, so
$\langle\, |[\ V : \nu \mid x\ ]|, m_V(\mu', \mu)\,\rangle \checkmark$.
$(m_V(\mu, \nu), \mu') \models d$, so
$(m_V(m_V(\mu, \nu), \mu), m_V(\mu', \mu)) \models d$ by lemma 40 , so
$(\mu, m_V(\mu', \mu)) \models d$, so
$\langle\, d \gg |[\ V : \nu \mid x\ ]|, \mu\,\rangle \checkmark$.

3. $\langle\, d \gg |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
   $\exists_{\mu''}\ (\mu, \mu'') \models d$ and $\langle\, |[\ V : \nu \mid x\ ]|, \mu''\,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
   $\exists_{w,p',w'}\ \langle\, x, m_V(\mu'', \nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p', w'\,\rangle$ with $l = m_V(w, \mu'')$, $p = |[\ V : w'\restriction_V \mid p'\ ]|$, $\mu' = m_V(w', \mu'')$
   $(m_V(\mu, \nu), m_V(\mu'', \nu)) \models d$ by lemma 40 , so
   $\langle\, d \gg x, m_V(\mu, \nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p', w'\,\rangle$, so
   $\langle\, |[\ V : \nu \mid d \gg x\ ]|, \mu\,\rangle \overset{a,m_V(w,\mu)}{\mapsto} \langle\, |[\ V : w'\restriction_V \mid p'\ ]|, m_V(w', \mu)\,\rangle$.
   $\mu''\restriction_V = \mu\restriction_V$ by lemma 49 , so
   $\langle\, |[\ V : \nu \mid d \gg x\ ]|, \mu\,\rangle \overset{a,m_V(w,\mu'')}{\mapsto} \langle\, p, m_V(w', \mu'')\,\rangle$, so
   $\langle\, |[\ V : \nu \mid d \gg x\ ]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu'\,\rangle$.
   Note that $(p, p) \in R$.
4. $\langle\, |[\ V : \nu \mid d \gg x\ ]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
   $\exists_{w,p',w'}\ \langle\, d \gg x, m_V(\mu, \nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p', w'\,\rangle$ with $l = m_V(w, \mu)$, $p = |[\ V : w'\restriction_V \mid p'\ ]|$, $\mu' = m_V(w', \mu)$, which needs the hypothesis
   $\exists_{\mu''}\ (m_V(\mu, \nu), \mu'') \models d$ and $\langle\, x, \mu''\,\rangle \overset{a,w}{\mapsto} \langle\, p', w'\,\rangle$.
   $\mu''\restriction_V = \nu$ by lemma 49 , so
   $\langle\, x, m_V(\mu'', \nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p', w'\,\rangle$, so
   $\langle\, x, m_V(m_V(\mu'', \mu), \nu)\,\rangle \overset{a,w}{\mapsto} \langle\, p', w'\,\rangle$, so
   $\langle\, |[\ V : \nu \mid x\ ]|, m_V(\mu'', \mu)\,\rangle \overset{a,m_V(w,m_V(\mu'',\mu))}{\mapsto} \langle\, |[\ V : w'\restriction_V \mid p'\ ]|, m_V(w', m_V(\mu'', \mu))\,\rangle$, so
   $\langle\, |[\ V : \nu \mid x\ ]|, m_V(\mu'', \mu)\,\rangle \overset{a,m_V(w,\mu)}{\mapsto} \langle\, |[\ V : w'\restriction_V \mid p'\ ]|, m_V(w', \mu)\,\rangle$, so
   $\langle\, |[\ V : \nu \mid x\ ]|, m_V(\mu'', \mu)\,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu'\,\rangle$.
   $(m_V(m_V(\mu, \nu), \mu), m_V(\mu'', \mu)) \models d$ by lemma 40 , so
   $(\mu, m_V(\mu'', \mu)) \models d$, so
   $\langle\, d \gg |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu'\,\rangle$.

5. $\langle\, d \gg |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
   $\exists_{\mu''}\ (\mu, \mu'') \models d$ and $\langle\, |[\ V : \nu \mid x\ ]|, \mu''\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
   $\exists_{\sigma,\sigma',p',w'}\ \langle\, x, m_V(\mu'', \nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$ with $s = m_V(\sigma, \sigma')$, $p = |[\ V : w'\restriction_V \mid p'\ ]|$, $\mu' = m_V(w', \sigma'(\uparrow))$.
   $(m_V(\mu, \nu), m_V(\mu'', \nu)) \models d$ by lemma 40 , so
   $\langle\, d \gg x, m_V(\mu, \nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$, so
   $\forall_{\sigma''}\ \langle\, |[\ V : \nu \mid d \gg x\ ]|, \mu\,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p'\ ]|, m_V(w', \sigma''(\uparrow))\,\rangle$.
   We take $\sigma'' = \sigma'$, so
   $\langle\, |[\ V : \nu \mid d \gg x\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$.
6. $\langle\, |[\ V : \nu \mid d \gg x\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
   $\exists_{\sigma,\sigma',p',w'}\ \langle\, d \gg x, m_V(\mu, \nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$ with $s = m_V(\sigma, \sigma')$, $p = |[\ V : w'\restriction_V \mid p'\ ]|$, $\mu' = m_V(w', \sigma'(\uparrow))$, which needs the hypothesis
   $\exists_{\mu''}\ (m_V(\mu, \nu), \mu'') \models d$ and $\langle\, x, \mu''\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$.
   $\mu''\restriction_V = \nu$ by lemma 49 , so
   $\langle\, x, m_V(\mu'', \nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$, so
   $\langle\, x, m_V(m_V(\mu'', \mu), \nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$, so
   $\forall_{\sigma''}\ \langle\, |[\ V : \nu \mid x\ ]|, m_V(\mu'', \mu)\,\rangle \overset{m_V(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\ V : w'\restriction_V \mid p'\ ]|, m_V(w', \sigma''(\uparrow))\,\rangle$.

We take $\sigma'' = \sigma'$, so
$\langle\, |[\ V : \nu \mid x\ ]|, m_V(\mu'', \mu)\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$.
$(m_V(m_V(\mu, \nu), \mu), m_V(\mu'', \mu)) \models d$ by lemma 40 , so
$(\mu, m_V(\mu'', \mu)) \models d$, so
$\langle\, d \gg |[\ V : \nu \mid x\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$.

The proof for the case of $(d \gg [\![\ V \mid x\ ]\!], [\![\ V \mid d \gg x\ ]\!]) \in R$ with $Var(d) \cap V = \emptyset$ is analogous to the case $(d \gg |[\ V : \nu \mid x\ ]|, |[\ V : \nu \mid d \gg x\ ]|) \in R$ with $Var(d) \cap V = \emptyset$. For $(x, x) \in R$ the proof is trivial.

Lemma 49 *If $(\mu, \mu') \models d$ and $Var(d) \cap V = \emptyset$ then $\mu\restriction_V = \mu'\restriction_V$, for any $\mu, \mu' \in Val$, $V \subseteq \mathcal{V}_{\mathrm{m}}$ and re-initialization clause d.*

Proof Suppose $(\mu, \mu') \models d$ and $Var(d) \cap V = \emptyset$. Suppose $d = [W \,|Pred\,]$ for some $W \subseteq \mathcal{V}_{\mathrm{m}}$ and re-initialization predicate *Pred*. Then, by the definition of re-initialization clauses, $\mu(n) = \mu'(n)$ for every $n \notin W$. $W \subseteq Var(d)$, so $W \cap V = \emptyset$. Therefore, $\mu(n) = \mu'(n)$ for every $n \in V$. We conclude that $\mu\restriction_V = \mu'\restriction_V$. ⊠

A.9 The Axiom: $[\![\ v \mid x\ ]\!] \approx [\![\ w \mid x[^w/_v]\ ]\!]$ if $w \notin Var(x)$

Take $R \subseteq \mathcal{T} \times \mathcal{T}$ to be the relation

$$\begin{array}{rcl} R & = & \{(|[\ v : \nu \mid x\ ]|, |[\ w : \nu' \mid x[^w/_v]\ ]|) \\ & & \mid v, w \in \mathcal{V}_{\mathrm{m}}, x \in \mathcal{T}, \nu, \nu' \in Val, w \notin Var(x), \nu(v) = \nu'(w)\} \\ & \cup & \{([\![\ v \mid x\ ]\!], [\![\ w \mid x[^w/_v]\ ]\!]) \mid v, w \in \mathcal{V}_{\mathrm{m}}, x \in \mathcal{T}, w \notin Var(x)\} \\ & \cup & \{(x, x) \mid x \in \mathcal{T}\} \end{array}$$

It is straightforward to see that this relation is a bisimulation relation.

B Miscellaneous Proofs

This appendix contains the proofs of several lemmas that are used in section 4.3.

B.1 Proof of lemma 21

We have to prove the following four statements

1. $[s \,|s^+ = C\,] \sim [s \,|s^+ = C'\,] \equiv [s \,|s^+ = C'\,]$
2. $d \sim [s \,|s^+ = C\,] \equiv [s \,|s^+ = C\,] \sim d$
3. $[s \,|s^+ = C\,] \equiv [s \,|\mathit{true}\,] \sim [s \,|s^+ = C\,]$
4. $[s \,|\mathit{pop}(s^-) \neq \emptyset \wedge s^+ = C\,] \equiv [\ \mathit{pop}(s^-) \neq \emptyset\ ] \sim [s \,|s^+ = C\,]$

for any $s \in \mathcal{V}_{\mathrm{m}}$, re-initialization clause d and constant expressions C, C' where $s \notin Var(d)$.

1. We prove this by showing that both re-initialization clauses accept the same set of solutions.

   Suppose $(\nu, \nu') \models [s\,|s^+ = C] \sim [s\,|s^+ = C']$. Then we know that $\nu(v) = \nu'(v)$ for all $v \notin \{s\}$ and that $\nu'(s) = C'$. Therefore, $(\nu, \nu') \models [s\,|s^+ = C']$.

   Suppose $(\nu, \nu') \models [s\,|s^+ = C']$. Then $\nu(v) = \nu'(v)$ for all $v \notin \{s\}$ and $\nu'(s) = C'$. Suppose $\nu''(v) = \nu(v)$ for all $v \notin \{s\}$ and that $\nu''(s) = C$. Then, $(\nu, \nu'') \models [s\,|s^+ = C]$ and $(\nu'', \nu') \models [s\,|s^+ = C']$. Therefore, $(\nu, \nu') \models [s\,|s^+ = C] \sim [s\,|s^+ = C']$.

   We conclude that $[s\,|s^+ = C] \sim [s\,|s^+ = C'] \equiv [s\,|s^+ = C']$.

2. We prove this by showing that both re-initialization clauses accept the same set of solutions.

   Suppose $(\nu, \nu') \models d \sim [s\,|s^+ = C]$. Then there is a ν'' such that $(\nu, \nu'') \models d$ and $(\nu'', \nu') \models [s\,|s^+ = C]$. Then, $\nu(v) = \nu'(v)$ and for any $v \notin \mathit{Var}(d) \cup \{s\}$, $\nu(v) = \nu''(v)$ and for any $v \notin \mathit{Var}(d)$, $\nu''(v) = \nu'(v)$ for any $v \in \mathit{Var}(d)$, and $\nu'(s) = C$.

   Suppose $\nu'''(s) = C$ and $\nu'''(v) = \nu(v)$ for any $v \notin \{s\}$. Then $(\nu, \nu''') \models [s\,|s^+ = C]$. Furthermore, $\nu'(v) = \nu(v)$ for any $v \notin \mathit{Var}(d) \cup \{s\}$, so $\nu'''(v) = \nu(v) = \nu'(v)$ for any $v \notin \mathit{Var}(d) \cup \{s\}$. $\nu'(s) = C = \nu'''(s)$, so $\nu'''(v) = \nu'(v)$ for any $v \notin \mathit{Var}(d)$. Now we can apply lemma 50, which gives $(\nu''', \nu') \models d$. We conclude that $(\nu, \nu') \models [s\,|s^+ = C] \sim d$.

   The proof for the second case is analogous to the proof for the previous case. We conclude that $d \sim [s\,|s^+ = C] \equiv [s\,|s^+ = C] \sim d$.

   Lemma 50 *If $(\nu, \nu'') \models d$, $\nu'''(n) = \nu(n)$ and $\nu'(n) = \nu''(n)$ for any $n \in \mathit{Var}(d)$ and $\nu'''(n) = \nu'(n)$ for any $n \notin \mathit{Var}(d)$, then $(\nu''', \nu') \models d$, for any re-initialization clause d and $\nu, \nu', \nu'', \nu''' \in \mathit{Val}$.*

   Proof Suppose $(\nu, \nu'') \models d$, $\nu'''(n) = \nu(n)$ and $\nu'(n) = \nu''(n)$ for any $n \in \mathit{Var}(d)$, and $\nu'''(n) = \nu'(n)$ for any $n \notin \mathit{Var}(d)$. Suppose d is of the form $[W\,|\mathit{Pred}]$, then $(\nu, \nu'') \models [W\,|\mathit{Pred}]$ implies that $\nu(n) = \nu''(n)$ for all $n \notin W$ and $(\nu, \nu'') \models \mathit{Pred}$ by definition of re-initialization clauses.

   This means that *Pred* evaluates to *true* for $n^- = \nu(n)$ and $n^+ = \nu''(n)$. Since only $n \in \mathit{Var}(d)$ occurs in *Pred*, *Pred* will still evaluate to *true* if we take arbitrary valuations for the variables $n \notin \mathit{Var}(d)$. Therefore, $(\nu''', \nu') \models \mathit{Pred}$.

   $\nu'''(n) = \nu(n)$ and $\nu'(n) = \nu''(n)$ for any $n \in \mathit{Var}(d)$ and $\nu(n) = \nu''(n)$ for all $n \notin W$, so $\nu'''(n) = \nu'(n)$ for all $n \in \mathit{Var}(d) \backslash W$. Furthermore, $\nu'''(n) = \nu'(n)$ for any $n \notin \mathit{Var}(d)$, so $\nu'''(n) = \nu'(n)$ for any $n \notin W$.

   $(\nu''', \nu') \models [W\,|\mathit{Pred}]$, so we conclude that $(\nu''', \nu') \models d$. ⊠

3. We prove this by showing that both re-initialization clauses accept the same set of solutions.

   Suppose $(\nu, \nu') \models [s\,|s^+ = C]$. $(\nu, \nu) \models [s\,|\mathit{true}]$, so $(\nu, \nu') \models [s\,|\mathit{true}] \sim [s\,|s^+ = C]$.

   Suppose $(\nu, \nu') \models [s\,|\mathit{true}] \sim [s\,|s^+ = C]$. Then there is a ν'' such that $(\nu, \nu'') \models [s\,|\mathit{true}]$ and $(\nu'', \nu') \models [s\,|s^+ = C]$. $\nu(v) = \nu''(v)$ and $\nu''(v) = \nu'(v)$ for any $v \notin \{s\}$ and $\nu'(s) = C$. Therefore, $\nu(v) = \nu'(v)$ for any $v \notin \{s\}$, so $(\nu, \nu') \models [s\,|s^+ = C]$.

   We conclude that $[s\,|s^+ = C] \equiv [s\,|\mathit{true}] \sim [s\,|s^+ = C]$

4. We prove this by a case distinction on $\mathit{pop}(s^-)$.

   - Suppose $\mathit{pop}(s^-) = \emptyset$.
     Then $[s\,|\mathit{pop}(s^-) \neq \emptyset \wedge s^+ = C] \equiv [s\,|\mathit{false} \wedge s^+ = C] \equiv [s\,|\mathit{false}] \equiv [\ \mathit{false}\ ]$ and $[\ \mathit{pop}(s^-) \neq \emptyset\ ] \sim [s\,|s^+ = C] \equiv [\ \mathit{false}\ ] \sim [s\,|s^+ = C] \equiv [\ \mathit{false}\ ]$.
     Therefore, $[s\,|\mathit{pop}(s^-) \neq \emptyset \wedge s^+ = C] \equiv [\ \mathit{pop}(s^-) \neq \emptyset\ ] \sim [s\,|s^+ = C]$.

- Suppose $pop(s^-) \neq \emptyset$. Then $[s\,|pop(s^-) \neq \emptyset \wedge s^+ = C] \equiv [s\,|true \wedge s^+ = C] \equiv [s\,|s^+ = C]$ and $\left[\ pop(s^-) \neq \emptyset\ \right] \sim [s\,|s^+ = C] \equiv \left[\ true\ \right] \sim [s\,|s^+ = C] \equiv [s\,|s^+ = C]$. Therefore, $[s\,|pop(s^-) \neq \emptyset \wedge s^+ = C] \equiv \left[\ pop(s^-) \neq \emptyset\ \right] \sim [s\,|s^+ = C]$.

We conclude that $[s\,|pop(s^-) \neq \emptyset \wedge s^+ = C] \equiv \left[\ pop(s^-) \neq \emptyset\ \right] \sim [s\,|s^+ = C]$.

B.2 Proof of lemma 22

We have to prove the following three statements

1. $[\![\ s\ \mid d_s \gg x\ ]\!] \approx x$
2. $[\![\ s\ \mid (d \sim d_s) \gg a \odot x\ ]\!] \approx [\![\ s\ \mid d \gg a \odot (d_s \gg x)\ ]\!]$
3. $[\![\ s\ \mid (d \sim d_s) \gg (c \wedge (s|\,\dot{s} = 0)) \rhd x\ ]\!] \approx [\![\ s\ \mid d \gg c \rhd (d_s \gg x)\ ]\!]$

for any $s \in \mathcal{V}_{\mathrm{m}}$, $a \in \mathcal{A}$, $x \in \mathcal{T}$, re-initialization clauses d, d_s and flow clause c, where $s \notin Var(c) \cup Var(x)$ and $d_s = [s\,|s^+ = C]$ with C a constant expression (i.e. it contains no model variables).

1. Let C be a constant expression. Take $R \subseteq \mathcal{T} \times \mathcal{T}$ to be the relation:

$$\begin{array}{rcl} R & = & \{(|[\ s : \nu \mid x\ ]|, |[\ s : \nu' \mid x\ ]|) \mid s \in \mathcal{V}_{\mathrm{m}}, x \in \mathcal{T}, s \notin Var(x), \nu, \nu' \in Val\} \\ & \cup & \{([\![\ s\ \mid d_s \gg x\ ]\!], [\![\ s\ \mid x\ ]\!]) \\ & & \mid s \in \mathcal{V}_{\mathrm{m}}, \text{re-init. clause } d_s, x \in \mathcal{T}, s \notin Var(x), d_s = [s\,|s^+ = C]\} \end{array}$$

   We omit the proof for the case $(|[\ s : \nu \mid x\ ]|, |[\ s : \nu' \mid x\ ]|) \in R$ with $s \notin Var(x)$ since it follows from the proof of soundness of axiom (VA6) and the fact that relation composition of two bisimulation relations is again a bisimulation relation.

   For $([\![\ s\ \mid d_s \gg x\ ]\!], [\![\ s\ \mid x\ ]\!]) \in R$ we have the following cases:

   (a) $\langle\, [\![\ s\ \mid d_s \gg x\ ]\!], \mu \,\rangle \checkmark$, which needs the hypothesis
   $\langle\, |[\ s : \nu \mid d_s \gg x\ ]|, \mu \,\rangle \checkmark$, which needs the hypothesis
   $\langle\, d_s \gg x, m_{\{s\}}(\mu, \nu) \,\rangle \checkmark$, which needs the hypothesis
   $\exists_{\mu'}\ (m_{\{s\}}(\mu, \nu), \mu') \models d_s$ and $\langle\, x, \mu' \,\rangle \checkmark$, so by lemma 42
   $\langle\, x, m_{\{s\}}(\mu', \nu') \,\rangle \checkmark$ for any ν'.
   $(m_{\{s\}}(\mu, \nu), m_{\{s\}}(\mu, \{s \mapsto C\})) \models d_s$, so
   $\langle\, x, m_{\{s\}}(m_{\{s\}}(\mu, \{s \mapsto C\}), \nu') \,\rangle \checkmark$, so
   $\langle\, x, m_{\{s\}}(\mu, \nu') \,\rangle \checkmark$, so
   $\langle\, |[\ s : \nu' \mid x\ ]|, \mu \,\rangle \checkmark$.

   (b) $\langle\, [\![\ s\ \mid x\ ]\!], \mu \,\rangle \checkmark$, which needs the hypothesis
   $\langle\, |[\ s : \nu' \mid x\ ]|, \mu \,\rangle \checkmark$, which needs the hypothesis
   $\langle\, x, m_{\{s\}}(\mu, \nu') \,\rangle \checkmark$, so by lemma 42
   $\langle\, x, \mu \,\rangle \checkmark$, so by lemma 42
   $\langle\, x, m_{\{s\}}(\mu, \{s \mapsto C\}) \,\rangle \checkmark$.
   $(m_{\{s\}}(\mu, \nu), m_{\{s\}}(\mu, \{s \mapsto C\})) \models d_s$ for any ν , so
   $\langle\, d_s \gg x, m_{\{s\}}(\mu, \nu) \,\rangle \checkmark$, so
   $\langle\, |[\ s : \nu \mid d_s \gg x\ ]|, \mu \,\rangle \checkmark$.

   (c) $\langle\, [\![\ s\ \mid d_s \gg x\ ]\!], \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
   $\langle\, |[\ s : \nu \mid d_s \gg x\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
   $\exists_{w,p',w'}\ \langle\, d_s \gg x, m_{\{s\}}(\mu, \nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$ with $l = m_{\{s\}}(w, \mu)$, $p = |[\ s : w' \restriction_{\{s\}} \mid p'\ ]|$, $\mu' = m_{\{s\}}(w', \mu)$, which needs the hypothesis

$\exists_{\mu''}\ (m_{\{s\}}(\mu,\nu),\mu'') \models d_s$ and $\langle\, x,\mu''\,\rangle \overset{a,w}{\mapsto} \langle\, p',w'\,\rangle$.
$(m_{\{s\}}(\mu,\nu), m_{\{s\}}(\mu,\{s\mapsto C\}) \models d_s$, so
$\langle\, x, m_{\{s\}}(\mu,\{s\mapsto C\})\,\rangle \overset{a,w}{\mapsto} \langle\, p',w'\,\rangle$, so by lemma 43
$w(s) = w'(s) = C$, so
$\langle\, x, m_{\{s\}}(\mu,\{s\mapsto C\})\,\rangle \overset{a,m_{\{s\}}(w,\{s\mapsto C\})}{\mapsto} \langle\, p', m_{\{s\}}(w',\{s\mapsto C\})\,\rangle$, so by lemma 44
$\langle\, x, m_{\{s\}}(\mu,\nu')\,\rangle \overset{a,m_{\{s\}}(w,\nu')}{\mapsto} \langle\, p', m_{\{s\}}(w',\nu')\,\rangle$ for any ν' , so
$\langle\, |[\, s : \nu' \mid x\,]|, \mu\,\rangle \overset{a,m_{\{s\}}(m_{\{s\}}(w,\nu'),\mu)}{\mapsto} \langle\, |[\, s : m_{\{s\}}(w',\nu')\restriction_{\{s\}} \mid p'\,]|, m_{\{s\}}(m_{\{s\}}(w',\nu'),\mu)\,\rangle$
, so
$\langle\, |[\, s : \nu' \mid x\,]|, \mu\,\rangle \overset{a,m_{\{s\}}(w,\mu)}{\mapsto} \langle\, |[\, s : \nu' \mid p'\,]|, m_{\{s\}}(w',\mu)\,\rangle$, so
$\langle\, |[\, s : \nu' \mid x\,]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, |[\, s : \nu' \mid p'\,]|, \mu'\,\rangle$, so
$\langle\, [\![\, s \mid x\,]\!], \mu\,\rangle \overset{a,l}{\mapsto} \langle\, |[\, s : \nu' \mid p'\,]|, \mu'\,\rangle$.
Note that $s \notin \mathit{Var}(p')$ by lemma 45, so $(|[\, s : w'\restriction_{\{s\}} \mid p'\,]|, |[\, s : \nu' \mid p'\,]|) \in R$.

(d) $\langle\, [\![\, s \mid x\,]\!], \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
$\langle\, |[\, s : \nu' \mid x\,]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
$\exists_{w,p',w'}\ \langle\, x, m_{\{s\}}(\mu,\nu')\,\rangle \overset{a,w}{\mapsto} \langle\, p',w'\,\rangle$ with $l = m_{\{s\}}(w,\mu)$, $p = |[\, s : w'\restriction_{\{s\}} \mid p'\,]|$, $\mu' = m_{\{s\}}(w',\mu)$, so by lemma 43
$\nu'(s) = w(s) = w'(s)$, so
$\langle\, x, m_{\{s\}}(\mu,\nu')\,\rangle \overset{a,m_{\{s\}}(w,\nu')}{\mapsto} \langle\, p', m_{\{s\}}(w',\nu')\,\rangle$, so by lemma 44
$\langle\, x, m_{\{s\}}(\mu,\{s\mapsto C\})\,\rangle \overset{a,m_{\{s\}}(w,\{s\mapsto C\})}{\mapsto} \langle\, p', m_{\{s\}}(w',\{s\mapsto C\})\,\rangle$.
$(m_{\{s\}}(\mu,\nu), m_{\{s\}}(\mu,\{s\mapsto C\}) \models d_s$ for any ν , so
$\langle\, d_s \gg x, m_{\{s\}}(\mu,\nu)\,\rangle \overset{a,m_{\{s\}}(w,\{s\mapsto C\})}{\mapsto} \langle\, p', m_{\{s\}}(w',\{s\mapsto C\})\,\rangle$, so
$\langle\, |[\, s : \nu \mid d_s \gg x\,]|, \mu\,\rangle \overset{a,m_{\{s\}}(m_{\{s\}}(w,\{s\mapsto C\}),\mu)}{\mapsto}$
$\langle\, |[\, s : m_{\{s\}}(w',\{s\mapsto C\})\restriction_{\{s\}} \mid p'\,]|, m_{\{s\}}(m_{\{s\}}(w',\{s\mapsto C\}),\mu)\,\rangle$, so
$\langle\, |[\, s : \nu \mid d_s \gg x\,]|, \mu\,\rangle \overset{a,m_{\{s\}}(w,\mu)}{\mapsto} \langle\, |[\, s : \{s\mapsto C\} \mid p'\,]|, m_{\{s\}}(w',\mu)\,\rangle$, so
$\langle\, |[\, s : \nu \mid d_s \gg x\,]|, \mu\,\rangle \overset{a,l}{\mapsto} \langle\, |[\, s : \{s\mapsto C\} \mid p'\,]|, \mu'\,\rangle$, so
$\langle\, [\![\, s \mid d_s \gg x\,]\!], \mu\,\rangle \overset{a,l}{\mapsto} \langle\, |[\, s : \{s\mapsto C\} \mid p'\,]|, \mu'\,\rangle$.
Note that $s \notin \mathit{Var}(p')$ by lemma 45, so $(|[\, s : \{s\mapsto C\} \mid p'\,]|, |[\, s : w'\restriction_{\{s\}} \mid p'\,]|) \in R$.

(e) $\langle\, [\![\, s \mid d_s \gg x\,]\!], \mu\,\rangle \overset{t}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
$\langle\, |[\, s : \nu \mid d_s \gg x\,]|, \mu\,\rangle \overset{t}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
$\exists_{\sigma,\sigma',p',w'}\ \langle\, d_s \gg x, m_{\{s\}}(\mu,\nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p',w'\,\rangle$ with $t = m_{\{s\}}(\sigma,\sigma')$, $p = |[\, s : w'\restriction_{\{s\}} \mid p'\,]|$, $\mu' = m_{\{s\}}(w',\sigma'(\uparrow))$, which needs the hypothesis
$\exists_{\mu''}\ (m_{\{s\}}(\mu,\nu),\mu'') \models d_s$ and $\langle\, x,\mu''\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p',w'\,\rangle$.
$(m_{\{s\}}(\mu,\nu), m_{\{s\}}(\mu,\{s\mapsto C\})) \models d_s$, so
$\langle\, x, m_{\{s\}}(\mu,\{s\mapsto C\})\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p',w'\,\rangle$, so by lemma 47
$w' = \sigma(\uparrow)$, so by lemma 48
$\langle\, x, m_{\{s\}}(\mu,\nu')\,\rangle \overset{m_{\{s\}}(\sigma,\sigma'')}{\rightsquigarrow} \langle\, p', m_{\{s\}}(w',\sigma''(\uparrow))\,\rangle$ for any ν',σ'' , so
$\forall_{\sigma'''}\ \langle\, |[\, s : \nu' \mid x\,]|, \mu\,\rangle \overset{m_{\{s\}}(m_{\{s\}}(\sigma,\sigma''),\sigma''')}{\rightsquigarrow}$
$\langle\, |[\, s : m_{\{s\}}(w',\sigma''(\uparrow))\restriction_{\{s\}} \mid p'\,]|, m_{\{s\}}(m_{\{s\}}(w',\sigma''(\uparrow)),\sigma'''(\uparrow))\,\rangle$.
We take $\sigma''' = \sigma'$, so
$\langle\, |[\, s : \nu' \mid x\,]|, \mu\,\rangle \overset{m_{\{s\}}(\sigma,\sigma')}{\rightsquigarrow} \langle\, |[\, s : \sigma''(\uparrow)\restriction_{\{s\}} \mid p'\,]|, m_{\{s\}}(w',\sigma'(\uparrow))\,\rangle$, so
$\langle\, |[\, s : \nu' \mid x\,]|, \mu\,\rangle \overset{t}{\rightsquigarrow} \langle\, |[\, s : \sigma''(\uparrow)\restriction_{\{s\}} \mid p'\,]|, \mu'\,\rangle$, so
$\langle\, [\![\, s \mid x\,]\!], \mu\,\rangle \overset{t}{\rightsquigarrow} \langle\, |[\, s : \sigma''(\uparrow)\restriction_{\{s\}} \mid p'\,]|, \mu'\,\rangle$.
Note that $s \notin \mathit{Var}(p')$ by lemma 45, so $(|[\, s : w'\restriction_{\{s\}} \mid p'\,]|, |[\, s : \sigma''(\uparrow)\restriction_{\{s\}} \mid p'\,]|) \in R$.

(f) $\langle\, [\![\, s \mid \nu'\,]\!]\, x, \mu\,\rangle \overset{t}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis

$\langle\, |[\, s : \nu' \mid x \,]|, \mu \,\rangle \overset{t}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
$\exists_{\sigma,\sigma',p',w'} \langle\, x, m_{\{s\}}(\mu, \nu') \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w' \,\rangle$ with $t = m_{\{s\}}(\sigma, \sigma')$, $p = |[\, s : w'\restriction_{\{s\}} \mid p' \,]|$, $\mu' = m_{\{s\}}(w', \sigma'(\uparrow))$, so by lemma 47
$w' = \sigma(\uparrow)$, so by lemma 48
$\langle\, x, m_{\{s\}}(\mu, \{s \mapsto C\}) \,\rangle \overset{m_{\{s\}}(\sigma,\sigma'')}{\rightsquigarrow} \langle\, p', m_{\{s\}}(w', \sigma''(\uparrow)) \,\rangle$ for any σ''.
$(m_{\{s\}}(\mu, \nu), m_{\{s\}}(\mu, \{s \mapsto C\})) \models d_s$ for any ν , so
$\langle\, d_s \gg x, m_{\{s\}}(\mu, \nu) \,\rangle \overset{m_{\{s\}}(\sigma,\sigma'')}{\rightsquigarrow} \langle\, p', m_{\{s\}}(w', \sigma''(\uparrow)) \,\rangle$, so
$\forall_{\sigma'''} \langle\, |[\, s : \nu \mid d_s \gg x \,]|, \mu \,\rangle \overset{m_{\{s\}}(m_{\{s\}}(\sigma,\sigma''),\sigma''')}{\rightsquigarrow}$
$\langle\, |[\, s : m_{\{s\}}(w', \sigma''(\uparrow))\restriction_{\{s\}} \mid p' \,]|, m_{\{s\}}(m_{\{s\}}(w', \sigma''(\uparrow)), \sigma'''(\uparrow)) \,\rangle$.
We take $\sigma''' = \sigma'$, so
$\langle\, |[\, s : \nu \mid d_s \gg x \,]|, \mu \,\rangle \overset{m_{\{s\}}(\sigma,\sigma')}{\rightsquigarrow} \langle\, |[\, s : \sigma''(\uparrow)\restriction_{\{s\}} \mid p' \,]|, m_{\{s\}}(w', \sigma'(\uparrow)) \,\rangle$, so
$\langle\, |[\, s : \nu \mid d_s \gg x \,]|, \mu \,\rangle \overset{t}{\rightsquigarrow} \langle\, |[\, s : \sigma''(\uparrow)\restriction_{\{s\}} \mid p' \,]|, \mu' \,\rangle$, so
$\langle\, |[\, s : \nu \mid d_s \gg x \,]|, \mu \,\rangle \overset{t}{\rightsquigarrow} \langle\, |[\, s : \sigma''(\uparrow)\restriction_{\{s\}} \mid p' \,]|, \mu' \,\rangle$.
Note that $s \notin \mathit{Var}(p')$ by lemma 45, so $(|[\, s : \sigma''(\uparrow)\restriction_{\{s\}} \mid p' \,]|, |[\, s : w'\restriction_{\{s\}} \mid p' \,]|) \in R$.

2. Let C be a constant expression. Take $R \subseteq \mathcal{T} \times \mathcal{T}$ to be the relation:

$$\begin{array}{rcl} R & = & \{(|[\, s : \nu \mid x \,]|, |[\, s : \nu' \mid x \,]|) \mid s \in \mathcal{V}_{\mathrm{m}}, x \in \mathcal{T}, s \notin \mathit{Var}(x), \nu, \nu' \in \mathit{Val}\} \\ & \cup & \{(|[\, s : \nu \mid \epsilon \odot x \,]|, |[\, s : \nu' \mid \epsilon \odot (d_s \gg x) \,]|) \\ & & \mid s \in \mathcal{V}_{\mathrm{m}}, \text{re-init. clause } d_s, x \in \mathcal{T}, s \notin \mathit{Var}(x), d_s = [s \,|s^+ = C\,]\} \\ & \cup & \{(\llbracket\, s \mid (d \sim d_s) \gg a \odot x \,\rrbracket, \llbracket\, s \mid d \gg a \odot (d_s \gg x) \,\rrbracket) \\ & & \mid s \in \mathcal{V}_{\mathrm{m}}, \text{re-init. clause } d, d_s, x \in \mathcal{T}, a \in \mathcal{A}, s \notin \mathit{Var}(x), d_s = [s \,|s^+ = C\,]\} \end{array}$$

For the case $(|[\, s : \nu \mid x \,]|, |[\, s : \nu' \mid x \,]|) \in R$, we refer to the proof of the previous item. We omit the proof for the case $(|[\, s : \nu \mid \epsilon \odot x \,]|, |[\, s : \nu' \mid \epsilon \odot (d_s \gg x) \,]|) \in R$, because we have soundness of the axiom $\epsilon \odot x \approx x$, congruence of bisimilarity, and the proof of lemma 22 (1).

For the case $(\llbracket\, s \mid (d \sim d_s) \gg a \odot x \,\rrbracket, \llbracket\, s \mid d \gg a \odot (d_s \gg x) \,\rrbracket) \in R$ we see that neither of the terms can terminate nor perform a flow transition. The following cases are left:

(a) $\langle\, \llbracket\, s \mid (d \sim d_s) \gg a \odot x \,\rrbracket, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
$\langle\, |[\, s : \nu \mid (d \sim d_s) \gg a \odot x \,]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
$\exists_{w,p',w'} \langle\, (d \sim d_s) \gg a \odot x, m_{\{s\}}(\mu, \nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$ with $l = m_{\{s\}}(w, \mu)$, $p = |[\, s : w'\restriction_{\{s\}} \mid p' \,]|$, $\mu' = m_{\{s\}}(w', \mu)$, which needs the hypothesis
$\exists_{\mu''} (m_{\{s\}}(\mu, \nu), \mu'') \models d \sim d_s$ and $\langle\, a \odot x, \mu'' \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$, which needs the hypothesis
$\exists_{p''} \langle\, a, \mu'' \,\rangle \overset{a,w}{\mapsto} \langle\, p'', w' \,\rangle$ with $p' = p'' \odot x$, which needs the hypothesis
$p'' = \epsilon$ and $\mu'' = w = w'$, so
$\langle\, a \odot (d_s \gg x), \mu'' \,\rangle \overset{a,w}{\mapsto} \langle\, \epsilon \odot (d_s \gg x), w' \,\rangle$.
$(\mu'', \mu'') \models d_s$, so
$(m_{\{s\}}(\mu, \nu), \mu'') \models d$, so
$\langle\, d \gg a \odot (d_s \gg x), m_{\{s\}}(\mu, \nu) \,\rangle \overset{a,w}{\mapsto} \langle\, \epsilon \odot (d_s \gg x), w' \,\rangle$, so
$\langle\, |[\, s : \nu \mid d \gg a \odot (d_s \gg x) \,]|, \mu \,\rangle \overset{a, m_{\{s\}}(w,\mu)}{\mapsto} \langle\, |[\, s : w'\restriction_{\{s\}} \mid \epsilon \odot (d_s \gg x) \,]|, m_{\{s\}}(w', \mu) \,\rangle$, so
$\langle\, |[\, s : \nu \mid d \gg a \odot (d_s \gg x) \,]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, |[\, s : w'\restriction_{\{s\}} \mid \epsilon \odot (d_s \gg x) \,]|, \mu' \,\rangle$, so
$\langle\, \llbracket\, s \mid d \gg a \odot (d_s \gg x) \,\rrbracket, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, |[\, s : w'\restriction_{\{s\}} \mid \epsilon \odot (d_s \gg x) \,]|, \mu' \,\rangle$.
Recall that $p = |[\, s : w'\restriction_{\{s\}} \mid p' \,]| = |[\, s : w'\restriction_{\{s\}} \mid \epsilon \odot x \,]|$ and note that $(|[\, s : w'\restriction_{\{s\}} \mid \epsilon \odot x \,]|, |[\, s : w'\restriction_{\{s\}} \mid \epsilon \odot (d_s \gg x) \,]|) \in R$.

(b) $\langle\, \llbracket\, s \mid d \gg a \odot (d_s \gg x) \,\rrbracket, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
$\langle\, |[\, s : \nu \mid d \gg a \odot (d_s \gg x) \,]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, p, \mu' \,\rangle$, which needs the hypothesis

$\exists_{w,p',w'} \langle\, d \gg a \odot (d_s \gg x), m_{\{s\}}(\mu,\nu) \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$ with $l = m_{\{s\}}(w,\mu),\ p = |[\ s : w'\restriction_{\{s\}} \ \mid p'\ ]|,\ \mu' = m_{\{s\}}(w',\mu)$, which needs the hypothesis
$\exists_{\mu''}\ (m_{\{s\}}(\mu,\nu),\mu'') \models d$ and $\langle\, a \odot (d_s \gg x), \mu'' \,\rangle \overset{a,w}{\mapsto} \langle\, p', w' \,\rangle$, which needs the hypothesis
$\exists_{p''} \langle\, a, \mu'' \,\rangle \overset{a,w}{\mapsto} \langle\, p'', w' \,\rangle$ with $p' = p'' \odot (d_s \gg x)$, which needs the hypothesis
$p'' = \epsilon$ and $\mu'' = w = w'$.
We take $\nu' = m_{\{s\}}(\mu'', \{s \mapsto C\})$, so
$(\mu'', \nu') \models d_s$, so
$(m_{\{s\}}(\mu,\nu),\nu') \models d \sim d_s$.
$\langle\, a, \nu' \,\rangle \overset{a,\nu'}{\mapsto} \langle\, \epsilon, \nu' \,\rangle$, so
$\langle\, a \odot x, \nu' \,\rangle \overset{a,\nu'}{\mapsto} \langle\, \epsilon \odot x, \nu' \,\rangle$, so
$\langle\, (d \sim d_s) \gg a \odot x, m_{\{s\}}(\mu,\nu) \,\rangle \overset{a,\nu'}{\mapsto} \langle\, \epsilon \odot x, \nu' \,\rangle$, so
$\langle\, |[\ s : \nu \mid (d \sim d_s) \gg a \odot x\ ]|, \mu \,\rangle \overset{a, m_{\{s\}}(\nu',\mu)}{\mapsto} \langle\, |[\ s : \nu'\restriction_{\{s\}} \ \mid \epsilon \odot x\ ]|, m_{\{s\}}(\nu',\mu) \,\rangle$.
$m_{\{s\}}(\nu',\mu) = m_{\{s\}}(\mu'',\mu) = m_{\{s\}}(w,\mu) = l$ and similarly, $m_{\{s\}}(\nu',\mu) = \mu'$, so
$\langle\, |[\ s : \nu \mid (d \sim d_s) \gg a \odot x\ ]|, \mu \,\rangle \overset{a,l}{\mapsto} \langle\, |[\ s : \nu'\restriction_{\{s\}} \ \mid \epsilon \odot x\ ]|, \mu' \,\rangle$. , so
$\langle\, [\![\ s \mid (d \sim d_s) \gg a \odot x\ ]\!], \mu \,\rangle \overset{a,l}{\mapsto} \langle\, |[\ s : \nu'\restriction_{\{s\}} \ \mid \epsilon \odot x\ ]|, \mu' \,\rangle$.
Recall that $p = |[\ s : w'\restriction_{\{s\}} \ \mid p'\ ]| = |[\ s : w'\restriction_{\{s\}} \ \mid \epsilon \odot (d_s \gg x)\ ]|$ and note that $(|[\ s : \nu'\restriction_{\{s\}} \ \mid \epsilon \odot x\ ]|, |[\ s : w'\restriction_{\{s\}} \ \mid \epsilon \odot (d_s \gg x)\ ]|) \in R$.

3. Let C be a constant expression. Take $R \subseteq \mathcal{T} \times \mathcal{T}$ to be the relation:

$$
\begin{array}{rcl}
R & = & \{(|[\ s : \nu \mid (c \wedge (\,s|\,\dot{s} = 0)) \blacktriangleright x\ ]|, |[\ s : \nu' \mid c \blacktriangleright (d_s \gg x)\ ]|) \\
 & & \mid s \in \mathcal{V}_{\mathrm{m}}, \nu, \nu' \in \mathit{Val}, \text{flow clause } c, \text{re-init. clause } d_s, x \in \mathcal{T}, \\
 & & s \notin \mathit{Var}(c) \cup \mathit{Var}(x), d_s = [s\,|s^+ = C], \nu(s) = C\} \\
 & \cup & \{([\![\ s \mid (d \sim d_s) \gg (c \wedge (\,s|\,\dot{s} = 0)) \rhd x\ ]\!], [\![\ s \mid d \gg c \rhd (d_s \gg x)\ ]\!]) \\
 & & \mid s \in \mathcal{V}_{\mathrm{m}}, \text{flow clause } c, \text{re-init. clause } d, d_s, x \in \mathcal{T}, \\
 & & s \notin \mathit{Var}(c) \cup \mathit{Var}(x), d_s = [s\,|s^+ = C]\} \\
 & \cup & \{(x, x) \mid x \in \mathcal{T}\}
\end{array}
$$

For the case $(|[\ s : \nu \mid (c \wedge (\,s|\,\dot{s} = 0)) \blacktriangleright x\ ]|, |[\ s : \nu' \mid c \blacktriangleright (d_s \gg x)\ ]|)$, we omit the proofs for termination and action transitions since they are trivial. The following cases are left:

(a) $\langle\, |[\ s : \nu \mid (c \wedge (\,s|\,\dot{s} = 0)) \blacktriangleright x\ ]|, \mu \,\rangle \overset{t}{\rightsquigarrow} \langle\, p, \mu' \,\rangle$, which needs the hypothesis
$\exists_{\sigma,\sigma',p',w'} \langle\, (c \wedge (\,s|\,\dot{s} = 0)) \blacktriangleright x, m_{\{s\}}(\mu,\nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w' \,\rangle$ with $t = m_{\{s\}}(\sigma,\sigma'),\ p = |[\ s : w'\restriction_{\{s\}} \ \mid p'\ ]|,\ \mu' = m_{\{s\}}(w',\sigma'(\uparrow))$, which needs one of the hypotheses

i. $\exists_{p''} \langle\, c \wedge (\,s|\,\dot{s} = 0), m_{\{s\}}(\mu,\nu) \,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'', w' \,\rangle$ with $p' = p'' \blacktriangleright x$, which needs the hypothesis
$(m_{\{s\}}(\mu,\nu),\sigma) \models c \wedge (\,s|\,\dot{s} = 0)$ and $w' = \sigma(\uparrow)$ and $p'' = c \wedge (\,s|\,\dot{s} = 0)$, so
$(m_{\{s\}}(\mu,\nu),\sigma) \models c$, so by lemma 46
$(m_{\{s\}}(m_{\{s\}}(\mu,\nu),\nu'), m_{\{s\}}(\sigma,\sigma'')) \models c$ for any ν', σ'' , so
$(m_{\{s\}}(\mu,\nu'), m_{\{s\}}(\sigma,\sigma'')) \models c$, so
$\langle\, c, m_{\{s\}}(\mu,\nu') \,\rangle \overset{m_{\{s\}}(\sigma,\sigma'')}{\rightsquigarrow} \langle\, c, m_{\{s\}}(\sigma(\uparrow),\sigma''(\uparrow)) \,\rangle$, so
$\langle\, c \blacktriangleright (d_s \gg x), m_{\{s\}}(\mu,\nu') \,\rangle \overset{m_{\{s\}}(\sigma,\sigma'')}{\rightsquigarrow} \langle\, c \blacktriangleright (d_s \gg x), m_{\{s\}}(\sigma(\uparrow),\sigma''(\uparrow)) \,\rangle$, so
$\forall_{\sigma'''} \langle\, |[\ s : \nu' \mid c \blacktriangleright (d_s \gg x)\ ]|, \mu \,\rangle \overset{m_{\{s\}}(m_{\{s\}}(\sigma,\sigma''),\sigma''')}{\rightsquigarrow} \langle\, |[\ s : m_{\{s\}}(\sigma(\uparrow),\sigma''(\uparrow))\restriction_{\{s\}} \mid c \blacktriangleright (d_s \gg x)\ ]|, m_{\{s\}}(m_{\{s\}}(\sigma(\uparrow),\sigma''(\uparrow)),\sigma'''(\uparrow)) \,\rangle$.
We take $\sigma''' = \sigma'$ and $\sigma'' = \sigma$, so
$\langle\, |[\ s : \nu' \mid c \blacktriangleright (d_s \gg x)\ ]|, \mu \,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ s : w'\restriction_{\{s\}} \ \mid c \blacktriangleright (d_s \gg x)\ ]|, \mu' \,\rangle$.
Recall that $p = |[\ s : w'\restriction_{\{s\}} \ \mid p'\ ]| = |[\ s : w'\restriction_{\{s\}} \ \mid p'' \blacktriangleright x\ ]| = |[\ s : w'\restriction_{\{s\}} \mid (c \wedge (\,s|\,\dot{s} = 0)) \blacktriangleright x\ ]|$ and note that $(|[\ s : w'\restriction_{\{s\}} \ \mid c \wedge (\,s|\,\dot{s} = 0) \blacktriangleright x\ ]|, |[\ s : w'\restriction_{\{s\}} \ \mid c \blacktriangleright (d_s \gg x)\ ]|) \in R$. Also note that $w'\restriction_{\{s\}}(s) = C$ necessarily.

ii. $\langle\, x, m_{\{s\}}(\mu,\nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$, so by lemma 47
$w' = \sigma(\uparrow)$, so by lemma 48
$\langle\, x, m_{\{s\}}(\mu,\{s \mapsto C\})\,\rangle \overset{m_{\{s\}}(\sigma,\sigma'')}{\rightsquigarrow} \langle\, p', m_{\{s\}}(w', \sigma''(\uparrow))\,\rangle$ for any σ''.
$(m_{\{s\}}(\mu,\nu'), m_{\{s\}}(\mu,\{s \mapsto C\})) \models d_s$ for any ν' , so
$\langle\, d_s \gg x, m_{\{s\}}(\mu,\nu')\,\rangle \overset{m_{\{s\}}(\sigma,\sigma'')}{\rightsquigarrow} \langle\, p', m_{\{s\}}(w', \sigma''(\uparrow))\,\rangle$, so
$\langle\, c \blacktriangleright (d_s \gg x), m_{\{s\}}(\mu,\nu')\,\rangle \overset{m_{\{s\}}(\sigma,\sigma'')}{\rightsquigarrow} \langle\, p', m_{\{s\}}(w', \sigma''(\uparrow))\,\rangle$, so
$\forall_{\sigma'''} \langle\, |[\ s : \nu' \mid c \blacktriangleright (d_s \gg x)\ ]|, \mu\,\rangle \overset{m_{\{s\}}(m_{\{s\}}(\sigma,\sigma''),\sigma''')}{\rightsquigarrow} \langle\, |[\ s : m_{\{s\}}(w', \sigma''(\uparrow))\restriction_{\{s\}} \mid p'\ ]|, m_{\{s\}}(m_{\{s\}}(w', \sigma''(\uparrow)), \sigma'''(\uparrow))\,\rangle$.
We take $\sigma''' = \sigma'$ and $\sigma'' = \sigma$, so
$\langle\, |[\ s : \nu' \mid c \blacktriangleright (d_s \gg x)\ ]|, \mu\,\rangle \overset{t}{\rightsquigarrow} \langle\, |[\ s : w'\restriction_{\{s\}} \mid p'\ ]|, \mu'\,\rangle$.
Note that $(|[\ s : w'\restriction_{\{s\}} \mid p'\ ]|, |[\ s : w'\restriction_{\{s\}} \mid p'\ ]|) \in R$.

(b) $\langle\, |[\ s : \nu' \mid c \blacktriangleright (d_s \gg x)\ ]|, \mu\,\rangle \overset{t}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
$\exists_{\sigma,\sigma',p',w'} \langle\, c \blacktriangleright (d_s \gg x), m_{\{s\}}(\mu,\nu')\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$ with $t = m_{\{s\}}(\sigma,\sigma')$, $p = |[\ s : w'\restriction_{\{s\}} \mid p'\ ]|$, $\mu' = m_{\{s\}}(w', \sigma'(\uparrow))$, which needs one of the hypotheses

i. $\exists_{p''} \langle\, c, m_{\{s\}}(\mu,\nu')\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'', w'\,\rangle$ with $p' = p'' \blacktriangleright (d_s \gg x)$, which needs the hypothesis
$(m_{\{s\}}(\mu,\nu'), \sigma) \models c$ and $w' = \sigma(\uparrow)$ and $p'' = c$.
We take $\sigma''(s)(t) = C$ for any t , then
$(m_{\{s\}}(\mu,\{s \mapsto C\}), m_{\{s\}}(\sigma,\sigma'')) \models (s|\, \dot{s} = 0)$ and by lemma 46 $(m_{\{s\}}(\mu,\{s \mapsto C\}), m_{\{s\}}(\sigma,\sigma'')) \models c$, so
$(m_{\{s\}}(\mu,\{s \mapsto C\}), m_{\{s\}}(\sigma,\sigma'')) \models c \wedge (s|\, \dot{s} = 0)$, so
$\langle\, c \wedge (s|\, \dot{s} = 0), m_{\{s\}}(\mu,\{s \mapsto C\})\,\rangle \overset{m_{\{s\}}(\sigma,\sigma'')}{\rightsquigarrow} \langle\, c \wedge (s|\, \dot{s} = 0), m_{\{s\}}(\sigma(\uparrow), \sigma''(\uparrow))\,\rangle$, so by lemma 48
$\langle\, c \wedge (s|\, \dot{s} = 0), m_{\{s\}}(\mu,\nu)\,\rangle \overset{m_{\{s\}}(\sigma,\sigma''')}{\rightsquigarrow}$
$\langle\, c \wedge (s|\, \dot{s} = 0), m_{\{s\}}(\sigma(\uparrow), \sigma'''(\uparrow))\,\rangle$ for any ν, σ''' , so
$\langle\, (c \wedge (s|\, \dot{s} = 0)) \blacktriangleright x, m_{\{s\}}(\mu,\nu)\,\rangle \overset{m_{\{s\}}(\sigma,\sigma''')}{\rightsquigarrow} \langle\, (c \wedge (s|\, \dot{s} = 0)) \blacktriangleright x, m_{\{s\}}(\sigma(\uparrow), \sigma'''(\uparrow))\,\rangle$, so
$\forall_{\sigma''''} \langle\, |[\ s : \nu \mid (c \wedge (s|\, \dot{s} = 0)) \blacktriangleright x\ ]|, \mu\,\rangle \overset{m_{\{s\}}(m_{\{s\}}(\sigma,\sigma'''),\sigma'''')}{\rightsquigarrow}$
$\langle\, |[\ s : m_{\{s\}}(\sigma(\uparrow), \sigma'''(\uparrow))\restriction_{\{s\}} \mid (c \wedge (s|\, \dot{s} = 0)) \blacktriangleright x\ ]|, m_{\{s\}}(m_{\{s\}}(\sigma(\uparrow), \sigma'''(\uparrow)), \sigma''''(\uparrow))\,\rangle$.
$\langle\, |[\ s : \nu \mid (c \wedge (s|\, \dot{s} = 0)) \blacktriangleright x\ ]|, \mu\,\rangle \overset{m_{\{s\}}(\sigma,\sigma'''')}{\rightsquigarrow} \langle\, |[\ s : \sigma'''(\uparrow)\restriction_{\{s\}} \mid (c \wedge (s|\, \dot{s} = 0)) \blacktriangleright x\ ]|, m_{\{s\}}(\sigma(\uparrow), \sigma''''(\uparrow))\,\rangle$.
We take $\sigma'''' = \sigma''' = \sigma'$ and $\nu(s) = C$, so
$\langle\, |[\ s : \nu \mid (c \wedge (s|\, \dot{s} = 0)) \blacktriangleright x\ ]|, \mu\,\rangle \overset{t}{\rightsquigarrow} \langle\, |[\ s : \sigma'(\uparrow)\restriction_{\{s\}} \mid (c \wedge (s|\, \dot{s} = 0)) \blacktriangleright x\ ]|, \mu'\,\rangle$.
Recall that $p = |[\ s : w'\restriction_{\{s\}} \mid p'\ ]| = |[\ s : w'\restriction_{\{s\}} \mid p'' \blacktriangleright (d_s \gg x)\ ]| = |[\ s : w'\restriction_{\{s\}} \mid c \blacktriangleright (d_s \gg x)\ ]|$ and note that $(|[\ s : \sigma'(\uparrow)\restriction_{\{s\}} \mid (c \wedge (s|\, \dot{s} = 0)) \blacktriangleright x\ ]|, |[\ s : w'\restriction_{\{s\}} \mid c \blacktriangleright (d_s \gg x)\ ]|) \in R$.

ii. $\langle\, d_s \gg x, m_{\{s\}}(\mu,\nu')\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$, which needs the hypothesis
$\exists_{\mu''} (m_{\{s\}}(\mu,\nu'), \mu'') \models d_s$ and $\langle\, x, \mu''\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$.
$(m_{\{s\}}(\mu,\nu'), m_{\{s\}}(\mu,\{s \mapsto C\})) \models d_s$, so
we can take $\mu'' = m_{\{s\}}(\mu,\{s \mapsto C\})$, so
$\langle\, x, m_{\{s\}}(\mu,\{s \mapsto C\})\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$, so by lemma 47
$w' = \sigma(\uparrow)$, so by lemma 48
$\langle\, x, m_{\{s\}}(\mu,\nu)\,\rangle \overset{m_{\{s\}}(\sigma,\sigma'')}{\rightsquigarrow} \langle\, p', m_{\{s\}}(w', \sigma''(\uparrow))\,\rangle$ for any ν, σ'' , so
$\langle\, (c \wedge (s|\, \dot{s} = 0)) \blacktriangleright x, m_{\{s\}}(\mu,\nu)\,\rangle \overset{m_{\{s\}}(\sigma,\sigma'')}{\rightsquigarrow} \langle\, p', m_{\{s\}}(w', \sigma''(\uparrow))\,\rangle$, so
$\forall_{\sigma'''} \langle\, |[\ s : \nu \mid (c \wedge (s|\, \dot{s} = 0)) \blacktriangleright x\ ]|, \mu\,\rangle \overset{m_{\{s\}}(m_{\{s\}}(\sigma,\sigma''),\sigma''')}{\rightsquigarrow}$

$\langle\, |[\, s : m_{\{s\}}(w', \sigma''(\uparrow))\restriction_{\{s\}} \mid p' \,]|, m_{\{s\}}(m_{\{s\}}(w', \sigma''(\uparrow)), \sigma'''(\uparrow))\,\rangle$.
We take $\sigma''' = \sigma'$ and $\sigma'' = \sigma$, so
$\langle\, |[\, s : \nu \mid (c \wedge (\, s|\, \dot{s} = 0)) \blacktriangleright x \,]|, \mu\,\rangle \overset{t}{\rightsquigarrow} \langle\, |[\, s : w'\restriction_{\{s\}} \mid p' \,]|, \mu'\,\rangle$.
Note that $\{(|[\, s : w'\restriction_{\{s\}} \mid p' \,]|, |[\, s : w'\restriction_{\{s\}} \mid p' \,]|)\} \in R$.

For the case $([\![\, s \mid (d \sim d_s) \gg (c \wedge (\, s|\, \dot{s} = 0)) \rhd x \,]\!], [\![\, s \mid d \gg c \rhd (d_s \gg x) \,]\!])$, we see that neither of the terms can terminate nor perform an action transition. The following cases are left:

(a) $\langle\, [\![\, s \mid (d \sim d_s) \gg (c \wedge (\, s|\, \dot{s} = 0)) \rhd x \,]\!], \mu\,\rangle \overset{t}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
$\langle\, |[\, s : \nu \mid (d \sim d_s) \gg (c \wedge (\, s|\, \dot{s} = 0)) \rhd x \,]|, \mu\,\rangle \overset{t}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
$\exists_{\sigma,\sigma',p',w'} \langle\, (d \sim d_s) \gg (c \wedge (\, s|\, \dot{s} = 0)) \rhd x, m_{\{s\}}(\mu, \nu)\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$ with $t = m_{\{s\}}(\sigma, \sigma')$, $p = |[\, s : w'\restriction_{\{s\}} \mid p' \,]|$, $\mu' = m_{\{s\}}(w', \sigma'(\uparrow))$, which needs the hypothesis
$\exists_{\mu''} (m_{\{s\}}(\mu, \nu), \mu'') \models d \sim d_s$ and $\langle\, (c \wedge (\, s|\, \dot{s} = 0)) \rhd x, \mu''\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$, which needs the hypothesis
$\exists_{p''} \langle\, c \wedge (\, s|\, \dot{s} = 0), \mu''\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'', w'\,\rangle$ with $p' = p'' \blacktriangleright x$, which needs the hypothesis
$(\mu'', \sigma) \models c \wedge (\, s|\, \dot{s} = 0)$ and $w' = \sigma(\uparrow)$ and $p'' = c \wedge (\, s|\, \dot{s} = 0)$, so
$(\mu'', \sigma) \models c$, so
$\langle\, c, \mu''\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, c, \sigma(\uparrow)\,\rangle$, so
$\langle\, c \rhd (d_s \gg x), \mu''\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, c \blacktriangleright (d_s \gg x), \sigma(\uparrow)\,\rangle$.
$(\mu'', \mu'') \models d_s$, so
$(m_{\{s\}}(\mu, \nu), \mu'') \models d$, so by lemma 40
$(m_{\{s\}}(\mu, \nu'), m_{\{s\}}(\mu'', \nu')) \models d$ for any ν' , so
there is a μ'' which is equal to $m_{\{s\}}(\mu'', \nu')$, so
$\langle\, c \rhd (d_s \gg x), m_{\{s\}}(\mu'', \nu')\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, c \blacktriangleright (d_s \gg x), \sigma(\uparrow)\,\rangle$, so
$\langle\, d \gg c \rhd (d_s \gg x), m_{\{s\}}(\mu, \nu')\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, c \blacktriangleright (d_s \gg x), \sigma(\uparrow)\,\rangle$, so
$\forall_{\sigma''} \langle\, |[\, s : \nu' \mid d \gg c \rhd (d_s \gg x) \,]|, \mu\,\rangle \overset{m_{\{s\}}(\sigma,\sigma'')}{\rightsquigarrow} \langle\, |[\, s : \sigma(\uparrow)\restriction_{\{s\}} \mid c \blacktriangleright (d_s \gg x) \,]|, m_{\{s\}}(\sigma(\uparrow), \sigma''(\uparrow))\,\rangle$.
We take $\sigma'' = \sigma'$, so
$\langle\, |[\, s : \nu' \mid d \gg c \rhd (d_s \gg x) \,]|, \mu\,\rangle \overset{t}{\rightsquigarrow} \langle\, |[\, s : w'\restriction_{\{s\}} \mid c \blacktriangleright (d_s \gg x) \,]|, \mu'\,\rangle$, so
$\langle\, [\![\, s \mid d \gg c \rhd (d_s \gg x) \,]\!], \mu\,\rangle \overset{t}{\rightsquigarrow} \langle\, |[\, s : w'\restriction_{\{s\}} \mid c \blacktriangleright (d_s \gg x) \,]|, \mu'\,\rangle$.
Recall that $p = |[\, s : w'\restriction_{\{s\}} \mid p' \,]| = |[\, s : w'\restriction_{\{s\}} \mid p'' \blacktriangleright x \,]| = |[\, s : w'\restriction_{\{s\}} \mid c \wedge (\, s|\, \dot{s} = 0) \blacktriangleright x \,]|$ and note that $(|[\, s : w'\restriction_{\{s\}} \mid c \wedge (\, s|\, \dot{s} = 0) \blacktriangleright x, |[\, s : w'\restriction_{\{s\}} \mid c \blacktriangleright (d_s \gg x) \,]|\,]|) \in R$. Note that $w'\restriction_{\{s\}}(s) = C$ necessarily.

(b) $\langle\, [\![\, s \mid d \gg c \rhd (d_s \gg x) \,]\!], \mu\,\rangle \overset{t}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
$\langle\, |[\, s : \nu' \mid d \gg c \rhd (d_s \gg x) \,]|, \mu\,\rangle \overset{t}{\rightsquigarrow} \langle\, p, \mu'\,\rangle$, which needs the hypothesis
$\exists_{\sigma,\sigma',p',w'} \langle\, d \gg c \rhd (d_s \gg x), m_{\{s\}}(\mu, \nu')\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$ with $t = m_{\{s\}}(\sigma, \sigma')$, $p = |[\, s : w'\restriction_{\{s\}} \mid p' \,]|$, $\mu' = m_{\{s\}}(w', \sigma'(\uparrow))$, which needs the hypothesis
$\exists_{\mu''} (m_{\{s\}}(\mu, \nu'), \mu'') \models d$ and $\langle\, c \rhd (d_s \gg x), \mu''\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p', w'\,\rangle$, which needs the hypothesis
$\exists_{p''} \langle\, c, \mu''\,\rangle \overset{\sigma}{\rightsquigarrow} \langle\, p'', w'\,\rangle$ with $p' = p'' \blacktriangleright (d_s \gg x)$, which needs the hypothesis
$(\mu'', \sigma) \models c$ and $w' = \sigma(\uparrow)$ and $p'' = c$.
We take $\sigma''(s)(t) = C$ for any t , then
$(m_{\{s\}}(\mu'', \{s \mapsto C\}), m_{\{s\}}(\sigma, \sigma'')) \models (\, s|\, \dot{s} = 0)$ and by lemma 46
$(m_{\{s\}}(\mu'', \{s \mapsto C\}), m_{\{s\}}(\sigma, \sigma'')) \models c$, so
$(m_{\{s\}}(\mu'', \{s \mapsto C\}), m_{\{s\}}(\sigma, \sigma'')) \models c \wedge (\, s|\, \dot{s} = 0)$, so
$\langle\, c \wedge (\, s|\, \dot{s} = 0), m_{\{s\}}(\mu'', \{s \mapsto C\})\,\rangle \overset{m_{\{s\}}(\sigma,\sigma'')}{\rightsquigarrow} \langle\, c \wedge (\, s|\, \dot{s} = 0), m_{\{s\}}(\sigma(\uparrow), \sigma''(\uparrow))\,\rangle$, so
$\langle\, (c \wedge (\, s|\, \dot{s} = 0)) \rhd x, m_{\{s\}}(\mu'', \{s \mapsto C\})\,\rangle \overset{m_{\{s\}}(\sigma,\sigma'')}{\rightsquigarrow} \langle\, (c \wedge (\, s|\, \dot{s} = 0)) \blacktriangleright x, m_{\{s\}}(\sigma(\uparrow), \sigma''(\uparrow))\,\rangle$.
$(m_{\{s\}}(\mu'', \nu), m_{\{s\}}(\mu'', \{s \mapsto C\})) \models d_s$ and by lemma 40 $(m_{\{s\}}(\mu, \nu), m_{\{s\}}(\mu'', \nu)) \models d$ for any ν , so

$(m_{\{s\}}(\mu,\nu), m_{\{s\}}(\mu'', \{s \mapsto C\})) \models d \sim d_s$, so

$\langle\, (d \sim d_s) \gg (c \wedge (\, s |\, \dot{s} = 0)) \;\rhd\; x, m_{\{s\}}(\mu,\nu)\,\rangle \overset{m_{\{s\}}(\sigma,\sigma'')}{\rightsquigarrow} \langle\, (c \wedge (\, s |\, \dot{s} = 0)) \blacktriangleright x, m_{\{s\}}(\sigma(\uparrow), \sigma''(\uparrow))\,\rangle$, so

$\forall_{\sigma'''}\ \langle\, |[\ s : \nu \mid (d \sim d_s) \gg (c \wedge (\, s |\, \dot{s} = 0)) \;\rhd\; x\ ]|, \mu\,\rangle \overset{m_{\{s\}}(m_{\{s\}}(\sigma,\sigma''),\sigma''')}{\rightsquigarrow} \langle\, |[\ s : m_{\{s\}}(\sigma(\uparrow), \sigma''(\uparrow))\restriction_{\{s\}} \mid (c \wedge (\, s |\, \dot{s} = 0)) \blacktriangleright x\ ]|, m_{\{s\}}(m_{\{s\}}(\sigma(\uparrow), \sigma''(\uparrow)), \sigma'''(\uparrow))\,\rangle$.
We take $\sigma''' = \sigma'$, so

$\langle\, |[\ s : \nu \mid (d \sim d_s) \gg (c \wedge (\, s |\, \dot{s} = 0)) \;\rhd\; x\ ]|, \mu\,\rangle \overset{m_{\{s\}}(\sigma,\sigma')}{\rightsquigarrow} \langle\, |[\ s : \sigma''(\uparrow)\restriction_{\{s\}} \mid (c \wedge (\, s |\, \dot{s} = 0)) \blacktriangleright x\ ]|, m_{\{s\}}(\sigma(\uparrow), \sigma'(\uparrow))\,\rangle$, so

$\langle\, |[\ s : \nu \mid (d \sim d_s) \gg (c \wedge (\, s |\, \dot{s} = 0)) \;\rhd\; x\ ]|, \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ s : \{s \mapsto C\} \mid (c \wedge (\, s |\, \dot{s} = 0)) \blacktriangleright x\ ]|, \mu'\,\rangle$,

$\langle\, [\![\ s \mid (d \sim d_s) \gg (c \wedge (\, s |\, \dot{s} = 0)) \;\rhd\; x\ ]\!], \mu\,\rangle \overset{s}{\rightsquigarrow} \langle\, |[\ s : \{s \mapsto C\} \mid (c \wedge (\, s |\, \dot{s} = 0)) \blacktriangleright x\ ]|, \mu'\,\rangle$.
Recall that $p = |[\ s : w'\restriction_{\{s\}} \mid p'\ ]| = |[\ s : w'\restriction_{\{s\}} \mid p'' \blacktriangleright (d_s \gg x)\ ]| = |[\ s : w'\restriction_{\{s\}} \mid c \blacktriangleright (d_s \gg x)\ ]|$ and note that $(|[\ s : \{s \mapsto C\} \mid (c \wedge (\, s |\, \dot{s} = 0)) \blacktriangleright x\ ]|, |[\ s : w'\restriction_{\{s\}} \mid c \blacktriangleright (d_s \gg x)\ ]|) \in R$.

For $(x, x) \in R$ the proof is trivial.