Syntax and Consistent Equation Semantics of Hybrid Chi [1]

D.A. van Beek [a], K.L. Man [b], M.A. Reniers [b], J.E. Rooda [a], R.R.H. Schiffelers [a]

[a] *Department of Mechanical Engineering, Eindhoven University of Technology (TU/e)*

[b] *Department of Mathematics and Computer Science, Eindhoven University of Technology (TU/e)*

Abstract

The hybrid χ (Chi) formalism integrates concepts from dynamics and control theory with concepts from computer science, in particular from process algebra and hybrid automata. It integrates ease of modeling with a straightforward, structured operational semantics. Its 'consistent equation semantics' enforces state changes to be consistent with delay predicates, that combine the invariant and flow clauses of hybrid automata. Ease of modeling is ensured by means of the following concepts: 1) different classes of variables: discrete and continuous, of subclass jumping or non-jumping, and algebraic; 2) strong time determinism of alternative composition in combination with delayable guards; 3) integration of urgent and non-urgent actions; 4) differential algebraic equations as a process term as in mathematics; 5) steady-state initialization; and 6) several user-friendly syntactic extensions. Furthermore, the χ formalism incorporates several concepts for complex system specification: 1) process terms for scoping that integrate abstraction, local variables, local channels and local recursion definitions; 2) process definition and instantiation that enable process re-use, encapsulation, hierarchical and/or modular composition of processes; and 3) different interaction mechanisms: handshake synchronization and synchronous communication that allow interaction between processes without sharing variables, and shared variables that enable modular composition of continuous-time or hybrid processes. The syntax and semantics are illustrated using several examples.

Email addresses: D.A.v.Beek@tue.nl (D.A. van Beek), K.L.Man@tue.nl (K.L. Man), M.A.Reniers@tue.nl (M.A. Reniers), J.E.Rooda@tue.nl (J.E. Rooda), R.R.H.Schiffelers@tue.nl (R.R.H. Schiffelers).

[1] Work partially done in the framework of the HYCON Network of Excellence, contract number FP6-IST-511368

1 Introduction

Hybrid systems related research is based on two, originally different, world views: on the one hand the dynamics and control (DC) world view, and on the other hand the computer science (CS) world view.

The DC world view is that of a predominantly continuous-time system, which is modeled by means of differential (algebraic) equations, or by means of a set of trajectories. Hybrid phenomena are modeled by means of discontinuous functions and/or switched equation systems. The evolution of a hybrid system in the continuous-time domain is considered as a set of piecewise continuous functions of time (one for each variable).

Analysis and synthesis of hybrid systems in the DC domain are done, among others, by means of piecewise affine (PWA) systems, mixed logic dynamical (MLD) systems or linear complementarity (LC) systems, see [1] for an overview relating these different classes, and see [2] for a translation of PWA systems to hybrid χ (Chi). A different framework to consider hybrid systems are differential (algebraic) equations with discontinuous right-hand sides, the semantics of which can be defined using differential inclusions. Such differential inclusions allow modeling of relays, valves or any kind of on/off switching elements at a high level of abstraction in control systems with so-called sliding modes [3,4].

The CS world view is that of a predominantly discrete-event system. A well-known model is a (hybrid) automaton, but modeling of discrete-event systems is also based on, among others, process algebra, Petri nets, and data flow languages. For modeling and analysis of hybrid phenomena, discrete-event formalisms are extended in different ways with some form of differential (algebraic) equations. The most influential hybrid system model is that of a hybrid automaton such as defined in [5–11]. An essential difference between such a hybrid automaton and a DC hybrid system model is that where in the DC hybrid model there are no actions, in the hybrid automaton, discontinuities take place mainly by means of (labeled) actions. By means of actions, the hybrid automaton switches from one mode/location to another mode/location.

Clearly, hybrid systems represent a domain where the DC and CS world views meet, and we believe that a formalism that integrates the DC and CS world views is a valuable contribution towards integration of the DC and CS methods, techniques, and tools. The hybrid χ formalism is such a formalism. On the one hand, it can deal with continuous-time systems, PWA/MLD/LC systems, and hybrid systems based on sets of ordinary differential equations using discontinuous functions in combination with algebraic constraints (the DC approach). On the other hand, it can deal with discrete-event systems, with-

out continuous variables or differential equations, and with hybrid systems in which discontinuities take place (mainly) by means of actions (the CS approach).

The intended use of hybrid χ is for modeling, simulation, verification, and real-time control. Its application domain ranges from physical phenomena, such as dry friction, to large and complex manufacturing systems. Although the semantics is formally defined, including a solution concept, the straightforward and elegant syntax and semantics is also highly suited to non-computer scientists. In the remainder of this paper, we usually refer to hybrid χ as χ.

The most important concepts in χ are summarized below:

(1) Integration between the DC and CS world views:
- The DC world view in χ allows modeling of hybrid phenomena by means of discontinuous functions and/or switched equation systems. For this purpose, χ has introduced the category of algebraic variables, the trajectory of which can be discontinuous. Furthermore, the convex equality operator, defined in [12], but not explained in detail in this paper, allows modeling of differential inclusions according to the Filippov solution concept [3]. The solution concept has been formalized in χ.
- The CS world view in χ allows modeling of hybrid phenomena in a way that is strongly influenced by hybrid automata. In this respect, the new hybrid χ formalism differs considerably from its predecessor defined in [13] which was quite different from hybrid automata. In the χ formalism described in this paper, the 'consistent equation semantics' enforces changes in the values of variables to be consistent with delay predicates, that combine the invariant and flow clauses of hybrid automata. This is expressed by the property $p \parallel x = e \leftrightarrows p[e/x] \parallel x = e$, that, although not yet proven, we expect to hold. Here, $\parallel$ denotes parallel composition, $x = e$ is a mathematical equation, $p[e/x]$ denotes the process term obtained from p by substituting every free occurrence of variable x by its defining expression e, and $p \leftrightarrows q$ means that the two process terms p and q are bisimilar, that is they have the same behavior. For example: $x := y \parallel y = 1$ is bisimilar to $x := 1 \parallel y = 1$, where $x := y$ denotes an assignment of the value of y to variable x. A difference between the consistent equation semantics and the semantics of hybrid automata is that where the χ semantics considers $\dot{x} = 1 \wedge \dot{x} = 2$ to be an inconsistent process term, the hybrid automaton can enter the location with flow clause $\dot{x} = 1 \wedge \dot{x} = 2$, but cannot delay in this location. The inconsistent process in a hybrid automaton is a location with invariant false. A translation from the hybrid automaton model defined in [8] to χ can be found in [2]. This translation assumes that the flow clauses of the hybrid automaton cannot evaluate to false.

(2) Integration of a straightforward semantics and ease of modeling.

An important aspect is the conceptual similarity with hybrid automata as mentioned in the previous item. The concepts from hybrid automata have been extended in several ways to facilitate modeling. Where hybrid automata in general either have locations (e.g. [6–8]) or discrete variables (e.g. [11]), and in addition either jumping or non-jumping continuous variables, χ has, among others, the following categories of variables:

- Discrete variables, which facilitate compact readable specifications. In hybrid automata such variables are sometimes mimicked by real valued variables with a derivative of zero. However, for non-real valued variables, such as variables of type string, the concept of a zero derivative cannot be used.
- Jumping continuous variables, that correspond to the continuous variables of hybrid automata as defined in, for example, [8]. The values of these variables are in principle allowed to jump (change) arbitrarily in an action transition, as long as the resulting values satisfy the action (jump) predicate, and the resulting process is consistent. Consider for example a system with three variables: x, y, z. If the value of x should change to 1, and the other variables should remain unchanged, the action (jump) predicate should be $x' = 1 \wedge y' = y \wedge z' = z$, or $x^+ = 1 \wedge y^+ = y^- \wedge z^+ = z^-$, depending on the syntax, where v' and v^+ denote the value of variable v after execution of the action, and v and v^- denote the value of variable v before execution of the action. Restrictions of the type $v^+ = v^-$ clutter the models, and are therefore often omitted in informal hybrid automata specifications. In order to allow fully formal models, without the clutter associated with the restrictions on non-jumping variables, χ has an additional class of variables: the non-jumping continuous variables.
- Non-jumping continuous variables, that correspond to the continuous variables of hybrid automata as defined by, for example, the input language of the tool HyTech [14]. The values of these variables are not allowed to change in action transitions, unless their changes are explicitly specified, for example by means of assigning a new value to such a variable.
- Algebraic variables, that can have discontinuous trajectories, as already discussed in the item on integration between the DC and CS world views.

Other concepts that enable integration of a straightforward semantics and ease of modeling are:

- Strong time-deterministic alternative composition operator. Where in many process algebras the passage of time can result in making a choice between the two operands of the choice or alternative composition operator, in χ, the passage of time can never result in such a choice. In the case of weak time-determinism, the alternative composition $\dot{x} = 1 \,[\!]\, x := 1$ (other languages may use the $+$ or $\oplus$ operators instead of $[\!]$) can non-deterministically choose between doing a delay

according to and resulting in $\dot{x} = 1$, or doing the (undelayable) action $x := 1$. Strong time deterministic alternative composition means that alternative composition can delay only if both process terms can delay together, so that $\dot{x} = 1 \,[\!]\, x := 1$ can only do the (non-delayable) action $x := 1$, and then terminate. Hybrid automata have a comparable choice mechanism, apart from initialization. In a hybrid automaton, action transitions cannot disappear as a result of time passing. They can only be disabled for the period of time that the associated guard evaluates to false in the valuation prescribed by the trajectory of the variables. Also, time passing cannot result in the choice of a different location. The only changes in a hybrid automaton as a result of time passing are changes in the values of the variables. Only initially, depending on the initial edges and invariants, different initial locations may be selected as a result of time passing. Note that this does not imply that the χ formalism (or a hybrid automaton) is time deterministic. In the case of equations with multiple solutions, such as in $x^2 = 1$, delaying can take place according to any of the allowed solutions.

- Delayable guards. Where many process algebras have non-delayable guards, χ has delayable guards. A non-delayable guard cannot perform a delay when it is false. A delayable guard can delay when it is false until it becomes true, and thus facilitates modeling. Consider for example a valve α that must be switched on when the temperature T becomes bigger then $T_{\max}$. Using a delayable guard, this can be modeled simply by $T \geq T_{\max} \rightarrow \alpha := \text{true}$.

  Delayable guards ensure that in $b \rightarrow h\,!\,b$, the value of expression b that is sent via channel h is always true. Note that $h\,!\,b$ can either do the send action, or delay for an arbitrary period of time. Non-delayable guards may lead to un-intuitive behavior, because the value of b that is sent may be false. Consider the process term:

$$x := 0;\ (\dot{x} = 1 \parallel (x \leq 3 \rightarrow h\,!\,x \,[\!]\, \Delta 10) \parallel \Delta 5;\ h\,?\,y),$$

  where $\Delta\, s$ can delay for t time-units ($t \leq s$) to $\Delta\, s - t$, and $\Delta\, 0$ can terminate by means of an internal action.

  Using non-delayable guards, the process term can perform the assignment, followed by a delay of at most 5, and after an internal action transforms into

$$\dot{x} = 1 \parallel (h\,!\,x \,[\!]\, \Delta 5) \parallel h\,?\,y.$$

  The guard that was true has disappeared at the start of the delay. If the communication via channel h takes place now, a value of 5 is sent, which does not conform to $x \leq 3$.

  Using delayable guards on the other hand, the process term can do the assignment followed by a delay of at most 3, and transforms into:

$$\dot{x} = 1 \parallel (x \leq 3 \rightarrow h\,!\,x \,[\!]\, \Delta 7) \parallel \Delta 2;\ h\,?\,y,$$

where the value of x is 3. Communication is still not possible. After a delay of 2, followed by an internal action, the process term transforms into:

$$\dot{x} = 1 \parallel (x \leq 3 \rightarrow h\,!\,x \,[\!]\, \Delta 5) \parallel h\,?\,y,$$

where the value of x is 5, and after another delay of 5 it transforms into:

$$\dot{x} = 1 \parallel (x \leq 3 \rightarrow h\,!\,x \,[\!]\, \Delta 0) \parallel h\,?\,y.$$

The time-out takes place, leading to: $\dot{x} = 1 \parallel h\,?\,y$. Due to the delayable guard, that does not disappear while delaying, the communication does not take place, because the guard cannot be satisfied.

- Integrated urgent and non-urgent actions. Where most hybrid automata have non-urgent actions only, the χ formalism has both non-urgent actions and urgent actions. The concept of urgency is defined in a very flexible way: non-delayable actions are by definition urgent and delayable actions are non-urgent. This is achieved without any additional operators. The concept of urgency is built into the individual parallel composition operator, alternative composition operator, and guard operator. Consider the non-delayable action $x := 1$. The following three process terms
  - $\dot{x} = 1 \parallel x := 1$
  - $\dot{x} = 1 \,[\!]\, x := 1$
  - $\dot{x} = 1 \parallel x \leq 0 \rightarrow x := 1$

  can each execute only the action $x := 1$, assuming that the value of x is initially non-positive. Consider now the delayable action $[x := 1]$. The following three process terms
  - $\dot{x} = 1 \parallel [x := 1]$
  - $\dot{x} = 1 \,[\!]\, [x := 1]$
  - $\dot{x} = 1 \parallel x \leq 0 \rightarrow [x := 1]$

  can each execute either the action $x := 1$ or perform a delay, assuming again that the value of x is initially non-positive.

  Communication on channels can also be urgent and non-urgent as in Uppaal. This is achieved by means of an operator that partitions the set of channels into a set of urgent and a set of non-urgent channels. For the urgent channels, communication must take place as soon as it becomes possible, whereas for the non-urgent channels, no such preference for communication is assumed (see Section 2.4.8).
- Non-causal equations as in mathematics. Differential algebraic equations are process terms in hybrid χ. Therefore, they are modeled in χ in the same way as in mathematics.
- Steady state initialization. Dynamical analysis of physical systems often starts in initial steady-state conditions. This means that the initial state is such that all derivatives are zero. In χ, steady state initialization can be easily expressed by means of the signal emission operator. For example, $\dot{x} = 0 \curvearrowright \dot{x} = -x + 1$ represents the steady state initialization

($\dot{x} = 0$) of model $\dot{x} = -x + 1$. This means that this model only allows behavior for the case that initially $\dot{x} = 0$ holds, which implies that the initial value of x must be 1. In general, steady state initialization is not possible in this way for hybrid automata, because initial edges and invariants are usually predicates over variables, not derivatives. However, when the equations are straightforward enough, the modeler can explicitly calculate steady state conditions. In the example, variable x could be initialized to 1.

- Syntactic extensions. Ease of modeling is further supported in χ by extension of the set of orthogonal core process terms with additional process terms for ease of modeling. These additional process terms are defined by means of a straightforward translation into the core process terms.

(3) Concepts for complex system specification:

- Process terms for scoping that integrate abstraction, local variables, local channels and local recursion definitions.
- Parameterized process definition and process instantiation that enable:
  - process re-use, and
  - encapsulation, hierarchical and/or modular composition of processes.
- CSP communication and synchronization concepts that allow synchronization and communication without sharing of variables.
- Shared variables, that enable modular composition of continuous or hybrid processes.

The history of the χ formalism dates back quite some time. It was originally designed as a modeling and simulation language for specification of discrete-event, continuous-time or combined discrete-event/continuous-time models. The first simulator [15], however, was suited to discrete-event models only. The simulator was successfully applied to a large number of industrial cases, such as an integrated circuit manufacturing plant, a brewery, and process industry plants [16]. Later, the hybrid language and simulator were developed [17,18]. For the purpose of verification, the discrete-event part of the language was mapped onto the process algebra χ_σ by means of a syntactical translation. The semantics of χ_σ was defined using a structured operational semantics style (SOS), bisimulation relations were derived, and a model checker was built [19]. In this way, verification of discrete-event χ models was made possible [20]. The χ formalism defined in this paper integrates the modeling language and the verification formalism. It integrates, extends and improves the syntax and semantics defined in [21] and [13].

The remainder of this paper is organized as follows. Section 2 describes the syntax and informal semantics of the χ formalism. In Section 3, the semantics of χ is formally specified. Several examples in Section 4 illustrate the use of the formalism. In Section 5, a notion of equivalence is defined, which is shown

to be a congruence for all χ operators. Furthermore, some useful properties of closed χ process terms are given. Full proofs are presented in the appendices. Section 6 discusses related work, and Section 7 terminates with conclusions and points out directions of future work.

2 Syntax and informal semantics of the Chi formalism

This section presents a concise definition of the syntax and informal semantics of χ. The syntax definition is incomplete in the sense that the syntax of predicates, expressions, etc. is defined on a high level of abstraction. This is done because different implementations of χ, such as tools for simulation, verification, or real-time control, may impose different syntactical restrictions. The intention of this paper is to define the χ formalism that encompasses all different future tools without posing unnecessary syntactical restrictions.

2.1 Syntax of processes

A χ process is a triple $\langle p, \sigma, E\rangle$, where p denotes a process term, σ denotes a valuation, and E denotes an environment. The syntax of process terms is introduced in Section 2.3. A valuation is a partial function from variables to values. Syntactically, a valuation is denoted by a set of pairs $\{x_0 \mapsto c_0, \ldots, x_n \mapsto c_n\}$, where x_i denotes a variable and c_i its value.

An environment E is a tuple (C, J, L, H, R), where C denotes the set of continuous variables, J denotes the set of jumping variables, L denotes the set of algebraic variables, H denotes the set of channels, and R denotes a recursion definition. A recursion definition is a partial function from recursion variables to process terms. Syntactically, a recursive process definition is denoted by a set of pairs $\{X_0 \mapsto p_0, \ldots, X_m \mapsto p_m\}$, where X_i denotes a recursion variable and p_i the process term defining it.

To ensure that the variables, channels and recursion variables occurring in χ processes are defined, each χ process $\langle p, \sigma, (C, J, L, H, R)\rangle$ must satisfy the following requirements:

- All variables occurring free in p or in the range of R must be either in the domain of σ, in set L, or in case of dotted variables $\dot{x}$, their undotted counterparts x must be in C.
- All channels occurring free in p or in the range of R must be in H.
- All recursion variables occurring free in p or in the range of R must be in the domain of R.

- The predefined variable time must be in the domain of σ, and not in any of the sets C, J, and L.
- Finally, continuous variables must have a value: $C \subseteq \mathrm{dom}(\sigma) \setminus \{\mathsf{time}\}$, jumping variables must be defined: $J \subseteq (\mathrm{dom}(\sigma) \setminus \{\mathsf{time}\}) \cup L$, and algebraic variables, recursion variables and the other variables must be disjoint: $\mathrm{dom}(\sigma) \cap L = \emptyset$ and $(\mathrm{dom}(\sigma) \cup L) \cap \mathrm{dom}(R) = \emptyset$.

2.2 Informal semantics of processes

The behavior of χ processes is defined in terms of actions and delays. Actions define instantaneous changes, where time does not change, to the values of variables. Delays involve the passing of time, where for all variables their trajectory as a function of time is defined. The valuation σ and the environment E, together define the variables that exist in the χ process and the variable classes to which they belong.

The variables are grouped into different classes with respect to the delay behavior and action behavior. With respect to the delay behavior, the variables are divided into the following classes:

- The discrete variables, the values of which remain constant while delaying.
- The continuous variables, the values of which change according to an absolutely continuous function of time while delaying. The values of continuous variables are further restricted by delay predicates, that are usually in the form of differential algebraic equations.
- The dotted continuous variables, the values of which change according to an integrable, possibly discontinuous function of time while delaying. The relation between the dotted continuous variables and the continuous variables is explained in Section 3.3.2.
- The algebraic variables, that behave in a similar way as continuous variables. The differences are that algebraic variables may change according to a discontinuous function of time, and algebraic variables are not allowed to occur as dotted variables.
- The predefined variable ‘time’, that denotes the current time.

With respect to the action behavior, the variables are divided into two classes:

- The non-jumping variables, the values of which by default do not change in actions. Such changes need to be explicitly specified.
- The jumping variables, the values of which by default can jump to arbitrary values in actions. The values after jumping can be restricted by means of action predicates, send and receive process terms, or delay predicates (equations).

The discrete and continuous variable classes can be divided into jumping and non-jumping versions. For the other classes, such a division is not possible: the dotted continuous and algebraic variables are by definition jumping with respect to the action behavior, and the predefined variable time is by definition non-jumping.

Further explanation on the semantics of the behavior of the different classes of variables is found in Section 3.3.1 on the action predicate, in Section 3.3.2 on the delay predicate, in Section 3.3.3 on the send and receive process terms, and in Section 3.4.6 on parallel composition.

The valuation σ captures the values of those variables that are relevant for determining the future behaviors of a process. The domain of the valuation σ in a χ process $\langle p, \sigma, E\rangle$ consists of the discrete variables, the continuous variables, and the predefined non-jumping variable time. The dotted continuous variables and the algebraic variables are not included in the domain of σ, because their values depend only on the process term p, possibly together with the values of the other variables. The values of the dotted continuous and algebraic variables are included in the so called 'extended valuation'. This extended valuation is required, among others, to ensure consistency of χ processes.

The consistency requirement enforces constraints on χ processes comparable to invariants in hybrid automata. Informally, in χ, the delay predicates (equations) must always hold. Consistency ensures that in $x := 1 \parallel y = x$, assuming that y is a jumping variable, the values of x and y are 1 after assigning 1 to x, independently of the initial value of y. Consistency also ensures that inconsistent processes cannot be reached, e.g. in $x := 1 \parallel x = 2$, the assignment to x cannot be executed. In fact, in χ, only consistent processes can perform action or delay transitions, and the result of an action or delay transition is always a consistent process.

Consistency is related to extended valuations in the following way: a χ process $\langle p, \sigma, E\rangle$ is consistent with extended valuation ξ, where ξ is the valuation σ extended with the (valuation for the) algebraic and dotted variables as defined by environment E, if the delay predicates u in p and the predicates u of signal emission operators in p hold when evaluated in extended valuation ξ.

For a χ process $\langle p, \sigma, (C, J, L, H, R)\rangle$, the combination of the variable classes for the delay and action behavior leads to the following classes of variables:

- The set of discrete variables D is $\mathrm{dom}(\sigma) \setminus (C \cup \{\mathsf{time}\})$,
  - the set of non-jumping discrete variables is $D \setminus J$,
  - the set of jumping discrete variables is $D \cap J$.
- The set of continuous variables is C,
  - the set of non-jumping continuous variables is $C \setminus J$,
  - the set of jumping continuous variables is $C \cap J$.

- The set of (jumping) dotted continuous variables is $\dot{C}$.
- The set of (jumping) algebraic variables is L.
- The predefined (non-jumping) variable denoting the current time is time.

Note that it is possible to have $D \cap J \neq \emptyset$ and $L \cap J \neq \emptyset$. Such jumping discrete or jumping algebraic variables can occur as an artefact of the parallel composition of a send and a receive process term, where the receive process term assigns the received value to a discrete or algebraic variable, see Sections 3.3.3 and 3.4.6. From a modeling perspective, discrete and algebraic variables are in principle never explicitly declared as jumping. Discrete variables are not declared as jumping, because their value is not determined by equations, and therefore their values need not change when the value of a variable occurring in an equation changes due to an action. Algebraic variables are not declared as jumping, because they are by definition jumping. In fact, there is no difference between the behavior of an algebraic variable that is in set J and one that is not in the set.

Consider, for example, the process $\langle\, n := 1 \parallel y = n,\ \{n \mapsto 0, \mathsf{time} \mapsto 0\},\ (\emptyset, \emptyset, \{y\}, \emptyset, \emptyset)\,\rangle$ consisting of the discrete variable n, the predefined variable time, the algebraic variable y, and no continuous variables. Initially, the value of n equals 0, and thus the value of y equals 0. After the assignment of 1 to n, the equation $y = n$ should still hold, and thus the value of y jumps to 1. The process terms and operators used in this model, and their informal semantics are discussed in the Sections 2.3 and 2.4.

2.3 Syntax of process terms

Process terms $p \in P$ (without $p_{\text{ext}} \in P_{\text{ext}}$, see the table below) are the 'core' elements of the χ formalism. In Section 2.5, the syntax of χ process terms is extended with process terms P_{ext} to ensure better readability of χ models. The semantics of those process terms is defined in terms of the core process terms given in this section.

The set of process terms P is defined by the following grammar for the process terms $p \in P$:

$$
\begin{aligned}
p ::= {} & W : r \gg l_{\mathrm{a}} \quad | \quad u \quad | \quad \delta \quad | \quad \bot \\
& | \; [p] \quad | \quad u \curvearrowright p \quad | \quad p;\, p \quad | \quad b \rightarrow p \quad | \quad p \,[\!]\, p \\
& | \; p \parallel p \quad | \quad h \,!!\, \mathbf{e}_n \quad | \quad h \,??\, \mathbf{x}_n \quad | \quad \partial_A(p) \quad | \quad \upsilon_H(p) \\
& | \; X \quad | \quad \iota_{J^+}(p) \\
& | \; [\![_{\mathrm{V}}\, \sigma_\bot, C, L \; `|' p \,]\!] \quad | \quad [\![_{\mathrm{H}}\, H \; `|' p \,]\!] \quad | \quad [\![_{\mathrm{R}}\, R \; `|' p \,]\!] \\
& | \; p_{\mathrm{ext}}
\end{aligned}
$$

Here, r is a predicate over variables (including the variable time), dotted continuous variables, and '$^-$' superscripted variables (including the dotted variables, e.g. $\dot{x}$). The action label l_{a} is taken from a given set A_{label} which at least contains the special action label τ representing the internal or silent step. Furthermore, u and b are both predicates over variables (including the variable time) and dotted continuous variables; $\mathbf{e}_n$ denotes the expressions $e_1, \ldots, e_n$, and $\mathbf{x}_n$ denotes the (non-dotted) variables $x_1, \ldots, x_n$ such that $\mathsf{time} \notin \{\mathbf{x}_n\}$. For $n = 0$, $h \,!!\, \mathbf{e}_n$ and $h \,??\, \mathbf{x}_n$ denote $h \,!!$ and $h \,??$, respectively, where h is a channel. Finally, A is a set of actions, H is a set of channels, X is a recursion variable, R is a recursion definition as defined in Section 2.1, W, J^+, C, L are sets of (non-dotted) variables such that $\mathsf{time} \notin W$ and $\mathsf{time} \notin J^+$, and $\sigma_\bot$ is a valuation that also allows the undefined 'value' $\bot$. It is specified as $\{x_0 \mapsto c_0, \ldots, x_n \mapsto c_n\}$, where x_i denotes a variable and c_i a value or $\bot$.

As is common practice in mathematics, the comma in predicates denotes conjunction. E.g. u_1, u_2 denotes the predicate $u_1 \wedge u_2$. Also, both $e_1 \leq \dot{x} \leq e_2$ and $\dot{x} \in [e_1, e_2]$ can be used instead of $\dot{x} \geq e_1$, $\dot{x} \leq e_2$, and likewise for strict inequalities and open intervals.

The operators are listed in descending order of their binding strength as follows $\{\curvearrowright,\ \rightarrow\}$, $\{;\ ,\}$, $\{\parallel\ , [\!]\}$. The operators inside the braces have equal binding strength. In addition, operators of equal binding strength associate to the right, and parentheses may be used to group expressions. For example, $p;\, q;\, r$ means $p;\, (q;\, r)$. An informal, concise explanation of this syntax is given below. Section 3 gives a more detailed account of their meaning.

2.4 Informal semantics of process terms

Strictly speaking, a χ process term p cannot perform actions nor delays. Only the χ process $\langle p, \sigma, E \rangle$, that is obtained by adding a valuation and an environment to p, can, in principle, perform actions and delays. Therefore, when we informally refer to a process term that performs actions or delays, we refer to the process term together with a valuation and environment.

2.4.1 Manipulating the values of variables

In χ, there are several classes of variables, and there are several means to change the value of a variable, depending on the class of variable. The main means for changing the value of a variable are the action predicate, for instantaneous changes, and the delay predicate, for the changes of variables over time.

Action predicates An instantaneous change of the value of a discrete or continuous variable in χ is always connected to the execution of an action. In action predicates, the action is represented by a label. Other types of action are related to communication, which is treated below in the paragraph on parallelism. *Action predicate* $W : r \gg l_{\mathrm{a}}$ denotes instantaneous changes to the variables from set W, by means of an action labeled l_{a}, such that predicate r is satisfied. The predefined global variable time cannot be assigned. The non-jumping variables that are not mentioned in W remain unchanged, and the jumping variables, dotted continuous variables, and algebraic variables may obtain 'arbitrary' values, provided that the predicate r is satisfied and the process remains consistent.

A '$^-$' superscripted occurrence of a variable refers to the value of the variable in the extended valuation prior to execution of the action predicate, and a normal (un-superscripted) occurrence of a variable refers to the value of that variable in the extended valuation that results from the execution of the action predicate. A predicate r is satisfied if evaluating the '$^-$' superscripted variables in the original extended valuation and evaluating the normal occurrences of the variables in the obtained extended valuation means that the predicate is true. The reason to use an extended valuation for evaluating action predicate r is that in such predicates also algebraic and dotted continuous variables may be used. Note that it can be the case that different instantaneous changes satisfy the predicate, this may result in non-determinism.

Note that the (multi-)assignment is not a primitive in χ, as for example in [19]. This is because action predicates are more expressive than assignments. An assignment can be expressed as an action predicate (see Section 2.5.2), but not the other way around. Consider for example the action predicate $\{x\} : x \in [0, 1] \gg \tau$, that changes the value of x to a value in the interval $[0, 1]$. Also, the predicate of an action predicate may consist of a conjunction of implicit equations, e.g. $\{\mathbf{x}\} : f_1(\mathbf{x}^-, \mathbf{x}) = 0 \wedge \ldots \wedge f_n(\mathbf{x}^-, \mathbf{x}) = 0 \gg \tau$. The solution of such a system of equations, if present, need not always be expressible in an explicit form. The system may also have multiple solutions.

Delay predicates In principle, continuous and algebraic variables change arbitrarily over time when delaying, although, depending on the class of the variable, they may have to respect some continuity requirements, see Section 3.3.2 for more details. A *delay predicate* u, usually in the form of a differential algebraic equation, restricts the allowed behavior of the continuous and algebraic variables in such a way that the value of the predicate remains true over time. Delay predicates in the form of $x \geq e$, where x is a variable, e an expression, and instead of $\geq$, also $\leq, >, <$ can be used, are comparable to invariants in hybrid automata.

2.4.2 Deadlock and inconsistency

In χ, only consistent processes can do action or delay transitions, and the result of an action or delay transition is always a consistent process. Some process terms are consistent for certain valuations and inconsistent for other valuations. E.g. the delay predicate process term $x \geq 0$ is consistent for all values of x greater or equal to zero, and inconsistent otherwise. There are also process terms that are consistent or inconsistent for all valuations. The *inconsistent process term* $\bot$ is inconsistent for all valuations. It cannot perform any transition.

The *deadlock process term* δ cannot perform actions or delays. It is however consistent. Both process terms are needed for the specification of properties only.

2.4.3 Any delay operator

Besides the specification of delay by means of delay predicates, arbitrary delay can be described by means of the *any delay operator* $[p]$. The resulting behavior is such that arbitrary delays are allowed. When $[p]$ delays, p remains unchanged and its delay behavior is ignored. The action behavior of p remains unchanged in $[p]$.

2.4.4 Signal emission

Signal emission operator process term $u \curvearrowright p$ behaves as p for those extended valuations where u holds. The process term is inconsistent with extended valuations for which u does not hold.

2.4.5 Sequential composition

The *sequential composition* of process terms p and q behaves as process term p until p terminates, and then continues to behave as process term q.

2.4.6 Conditional

The *guarded process term* $b \rightarrow p$ can do whatever actions p can do under the condition that the guard b evaluates to true using the current extended valuation. The guarded process term can delay according to p under the condition that for the intermediate extended valuations during the delay, the guard b holds. The guarded process term can perform arbitrary delays under the condition that for the intermediate valuations during the delay, possibly excluding the first and last valuation, the guard b does not hold.

2.4.7 Choice

The *alternative composition operator* $[]$ allows a non-deterministic choice between different actions of a process. With respect to time behavior, the participants in the alternative composition have to synchronize. This means that the trajectories of the variables have to be agreed upon by both participants. This means that $[]$ is a strong time-deterministic choice operator.

2.4.8 Parallelism

Parallelism can be specified by means of the *parallel composition operator* $\|$. Parallel processes interact by means of shared variables or by means of synchronous point-to-point communication/synchronization via a channel. Channels are denoted as labels (identifiers). The parallel composition $p \,\|\, q$ synchronizes the time behavior of p and q, interleaves the action behavior (including the instantaneous changes of variables) of p and q, and synchronizes matching send and receive actions. The synchronization of time behavior means that only the time behaviors that are allowed by both p and q are allowed by their parallel composition. The consistent equation semantics of χ enforces that actions by p (or q) are allowed only if the values of the variables before and after the actions are consistent with the other process term q (or p). This means, among others, that the delay predicates of q must hold before and after execution of an action by p.

By means of the *send process term* $h \,!!\, e_1, \ldots, e_n$, for $n \geq 1$, the values of expressions $e_1, \ldots, e_n$ (evaluated w.r.t. the extended valuation) are sent via channel h. For $n = 0$, this reduces to $h \,!!$ and nothing is sent via the channel. By means of the *receive process term* $h \,??\, x_1, \ldots, x_n$, for $n \geq 1$, values for

$x_1, \ldots, x_n$ are received from channel h. We assume that all variables in $\mathbf{x}_n$ are different: $x_i = x_j \implies i = j$. For $n = 0$, this reduces to $h\,??$, and nothing is received via the channel. Communication in χ is the sending of values by one parallel process via a channel to another parallel process, where the received values (if any) are stored in variables. For communication, the acts of sending and receiving (values) have to take place in different parallel processes at the same moment in time. In case no values are sent and received, we refer to synchronization instead of communication.

In order to be able to model open systems (i.e. systems that interface with the environment), it is necessary not to enforce communication via the external channels of the model (e.g. the channels that send or receive from the environment). For communication via internal channels, however, the communication of matching send and receive actions, often is not only an option, but an obligation. In such models, the separate occurrence of the send action and the receive action via an internal channel is undesired. The *encapsulation operator* ∂_A, where $A \subseteq \mathcal{A} \setminus \{\tau\}$ is a set of actions ($\mathcal{A}$ is the set of all possible actions and τ is the predefined internal action), is introduced to block the actions from the set A. In order to assure that, for internal channels, only the synchronous execution of matching send and receive actions takes place, one can simply put all send and receive actions via internal channels in the set A.

In principle the channels in χ are non-urgent. This means that communication does not necessarily take place as soon as possible. In order to describe also urgent channels, the *urgent communication operator* $\upsilon_H(p)$, where $H \subseteq \mathcal{H}$ is a set of channel labels, ensures that p can only delay in case no communication or synchronization of send and receive actions via a channel from H is possible.

Note that a different kind of urgency can be achieved by means of undelayable process terms. The χ semantics ensures that actions of undelayable process terms have priority over delays. For example in $\dot{x} = 1 \parallel x := 1$ and $\dot{x} = 1 \,[\!]\, x := 1$, the assignment cannot delay. Therefore, it must be executed before a delay is possible. Also in $h\,!! \parallel \dot{x} = 1$, or $h\,!! \parallel h\,??$, or $h\,!! \parallel [h\,??]$, the parallel composition cannot delay because $h\,!!$ cannot delay. Therefore, a send action must be executed before a delay may be possible. Process term $[h\,!!] \parallel [h\,??]$, however, can do a communication action (or send or receive action), but it can also delay. To enforce the communication action to take place in such case, the urgent communication operator can be used: $\upsilon_{\{h\}}([h\,!!] \parallel [h\,??])$.

2.4.9 Recursive definitions

Process term X denotes a recursion variable (identifier) that is defined either in the environment of the process, or in a recursion scope operator process term $[\![_{\mathrm{R}} \ldots \mid p \,]\!]$, see below. Among others, it is used to model repetition.

Recursion variable X can do whatever the process term of its definition can do.

2.4.10 Jump enabling operator

Jump enabling operator $\iota_{J^+}(p)$, where J^+ denotes a set of variables, is used to (re)define the variables in set J^+ as jumping variables.

2.4.11 Hierarchical modeling

Thus far, it has been assumed that all variables that are allowed to occur in a χ process term are either declared in the valuation or in the environment (in the set L). To support the hierarchical modeling of systems, it is convenient to allow local declarations of variables. For this purpose, the *variable scope operator* process term $[\![_{\mathrm{V}}\, \sigma_\perp, C, L \mid p \,]\!]$ is introduced, where $\sigma_\perp$ denotes a valuation of local variables, where values may be undefined ($\perp$), C denotes a set of local (non-jumping) continuous variables, and L denotes a set of local algebraic variables. The set of local discrete variables is $\mathrm{dom}(\sigma_\perp) \setminus C$. We assume $C \subseteq \mathrm{dom}(\sigma_\perp)$, $\mathrm{dom}(\sigma_\perp) \cap L = \emptyset$, and $C \cap L = \emptyset$. It is allowed that the local variables have been declared on a more global level already. Any occurrence of a variable from $\mathrm{dom}(\sigma_\perp) \cup \dot{C} \cup L$ in process term p refers to the local variable and not to any more global declaration of the same variable name.

For similar purposes, local channels can be declared by means of a *channel scope* process term $[\![_{\mathrm{H}}\, H \mid p \,]\!]$, and local recursive definitions by means of a *recursion scope* process term $[\![_{\mathrm{R}}\, R \mid p \,]\!]$. The channel scope process term $[\![_{\mathrm{H}}\, H \mid p \,]\!]$ is used to declare the channels from the set $H \subseteq \mathcal{H}$ to be local. Communication actions via those local channels are abstracted from (replaced by internal action τ), and the separate send and receive actions via local channels are blocked. The recursion scope process term $[\![_{\mathrm{R}}\, R \mid p \,]\!]$ is used to declare local recursion definitions $R \subseteq \mathcal{R}$ (see Section 3.1 for the definition of $\mathcal{R}$).

2.5 Syntactic extensions

For many of the χ processes, process terms and operators introduced before, there is additional, more user-friendly syntax available, the so-called syntactic extensions. In this section, all of these syntactic extensions are expressed in terms of the syntax introduced in the previous sections.

2.5.1 Processes

Notation

$$
\begin{array}{l}
\langle\ \mathsf{disc}\ s_1, \ldots, s_k \\
,\ \mathsf{cont}\ x_1, \ldots, x_n \\
,\ \mathsf{alg}\ z_1, \ldots, z_m \\
,\ \mathsf{chan}\ h_1, \ldots, h_l \\
,\ i \\
,\ X_1 \mapsto p_1, \ldots, X_r \mapsto p_r \\
\mid\ p \\
\rangle,
\end{array}
$$

where $s_1, \ldots, s_k$ denote the discrete variables, $x_1, \ldots, x_n$ denote the non-jumping continuous variables, $z_1, \ldots, z_m$ denote the algebraic variables, $h_1, \ldots, h_l$ denote the urgent channels, i denotes an initialization predicate that restricts the allowed values of the variables initially, $X_1 \mapsto p_1, \ldots, X_r \mapsto p_r$ denote the recursion definitions, and p is a process term, is an abbreviation for the set of χ processes defined by:

$$
\begin{array}{l}
\langle\ \partial_{A_{\mathrm{ia}}}(\upsilon_{\{h_1,\ldots,h_l\}}((i \wedge \mathsf{time} = 0) \curvearrowright p)) \\
,\ \sigma_{sxt} \\
,\ (\ \{x_1, \ldots, x_n\} \\
\quad ,\ \emptyset \\
\quad ,\ \{z_1, \ldots, z_m\} \\
\quad ,\ \{h_1, \ldots, h_l\} \\
\quad ,\ \{X_1 \mapsto p_1, \ldots, X_r \mapsto p_r\} \\
\quad)\ \\
\rangle,
\end{array}
$$

namely for each valuation σ_{sxt}, with $\mathrm{dom}(\sigma_{sxt}) = \{s_1, \ldots, s_k, x_1, \ldots, x_n, \mathsf{time}\}$, a separate χ process. In the χ process, A_{ia} represents the internal send and receive actions via channels $h_1, \ldots, h_l$.

In the notation defined above, it is required that the discrete, continuous, and algebraic variables are all different. Besides the declared variables, the existence of the predefined reserved global variable time which denotes the current time, the value of which is initially zero, is assumed. This variable cannot be declared. It can only be used in expressions in process term p, or in $p_1, \ldots, p_r$.

As a shorthand, the keyword preceding variables of a certain type is omitted when there are no variables of that type, and the keyword chan is omitted

when there are no channel declarations. Also the initialization predicate i and the recursive definitions $X_1 \mapsto p_1, \ldots, X_r \mapsto p_r$ may be omitted, indicating a predicate that always holds and an empty list of recursive definitions, respectively.

2.5.2 Process terms

For many of the core process terms introduced before, there is additional, more user-friendly syntax available. The set of process terms P_{ext} is defined by the following grammar for the process terms $p_{\text{ext}} \in P_{\text{ext}}$ and $p \in P$:

$$\begin{aligned} p_{\text{ext}} ::= {} & \text{skip} \mid \mathbf{x}_n := \mathbf{e}_n \mid h\,!\,\mathbf{e}_n \mid h\,?\,\mathbf{x}_n \\ & \mid \Delta_d(p) \mid \Delta d \mid *p \mid b \xrightarrow{*} p \\ & \mid (\mathsf{jump}\ \mathbf{y}_m\ \text{`}|\text{'}\ p) \\ & \mid [\![\ \mathsf{disc}\ \mathbf{s}_k, \mathsf{cont}\ \mathbf{x}_n, \mathsf{alg}\ \mathbf{z}_l, \mathsf{chan}\ \mathbf{h}_m, i, L_R\ \text{`}|\text{'}\ p\ ]\!] \\ & \mid l_{\text{p}}(\mathbf{x}_k,\ \mathbf{h}_m,\ \mathbf{e}_n) \end{aligned}$$

The operators of p and p_{ext} are listed in descending order of their binding strength as follows: $\{*,\ \xrightarrow{*}, \curvearrowright,\ \rightarrow\},\ ;\ ,\ \{\|\ ,[\!]\}$.

Skip Process term skip is an abbreviation for an action predicate that can perform an internal action (τ), such that only the jumping variables can change.

$$\boxed{\text{skip} \triangleq \emptyset : \text{true} \gg \tau}$$

Multi-assignment Multi-assignment $\mathbf{x}_n := \mathbf{e}_n$ for $n \geq 1$ is an abbreviation for an internal action that changes variables $x_1, \ldots, x_n$ to the values of expressions $e_1, \ldots, e_n$, respectively. For $n = 1$, this gives an assignment $x := e$.

$$\boxed{\mathbf{x}_n := \mathbf{e}_n \triangleq \{\mathbf{x}_n\} : x_1 = e_1^- \wedge \cdots \wedge x_n = e_n^- \gg \tau}$$

Here e^- denotes the result of replacing all variables x_i in e by their '$^-$' superscripted version x_i^-. For example, process term $x := 2x + yz$ is defined as $\{x\} : x = 2x^- + y^- z^- \gg \tau$, and process term $x, y := x + y, x - y$ is defined as $\{x, y\} : (x = x^- + y^-) \wedge (y = x^- - y^-) \gg \tau$.

Delayable send and receive Process terms $h\,!\,\mathbf{e}_n$, and $h\,?\,\mathbf{x}_n$ are the respective delayable counterparts of $h\,!!\,\mathbf{e}_n$ and $h\,??\,\mathbf{x}_n$. They are defined by means of the any delay operator $[p]$, which adds arbitrary delay behavior to p.

$$h\,!\,\mathbf{e}_n \triangleq [h\,!!\,\mathbf{e}_n] \qquad h\,?\,\mathbf{x}_n \triangleq [h\,??\,\mathbf{x}_n]$$

Delay operators By means of the delay operator $\Delta_d(p)$, a process term is forced to delay for the amount of time units specified by the value of numerical expression d, and then proceeds as p. The abbreviation Δd denotes a process term that first delays for d time units, and then terminates by means of an internal action τ. The fact that process term Δd terminates by means of an action ensures that time-outs enforce a choice in alternative composition. The value of expression d is evaluated at the first delay or action by Δd.

$$\begin{aligned}\Delta_d(p) &\triangleq [\![_{\mathrm{V}}\, \{t \mapsto \bot\}, \emptyset, \emptyset \mid t = \mathsf{time} + d \curvearrowright \mathsf{time} \geq t \rightarrow p \,]\!] \\ \Delta d \quad &\triangleq \Delta_d(\mathrm{skip})\end{aligned}$$

In the definition of $\Delta_d(p)$, t denotes a fresh variable, not occurring free in p. Delays are only defined for non-negative values of d. Therefore, we assume that the value of d in the extended valuation is non-negative.

Repetition operators Process term $*p$ represents the infinite repetition of process term p. Guarded repetition $b \stackrel{*}{\rightarrow} p$ can be interpreted as 'while b do p'.

$$\begin{aligned}*p \quad &\triangleq [\![_{\mathrm{R}}\, \{X \mapsto p;\ X\} \mid X \,]\!] \\ b \stackrel{*}{\rightarrow} p &\triangleq [\![_{\mathrm{R}}\, \{X \mapsto b \rightarrow \mathrm{skip};\ p;\ X \,[\!]\, \neg b \rightarrow \mathrm{skip}\} \mid X \,]\!]\end{aligned}$$

In the definition of $*p$ and $b \stackrel{*}{\rightarrow} p$, recursion variable X denotes a fresh recursion variable not occurring free in p.

Jump enabling operator Jump enabling operator $(\mathsf{jump}\,\mathbf{y}_m \mid p)$, where $\mathbf{y}_m$ denotes a comma separated list of variables, is used to redefine the variables $\mathbf{y}_m$ as jumping variables.

$$(\mathsf{jump}\ \mathbf{y}_m \mid p) \triangleq \iota_{\{\mathbf{y}_m\}}(p)$$

Modeling scope operator The modeling scope operator process term

$$[\![\ \mathsf{disc}\ \mathbf{s}_k, \mathsf{cont}\ \mathbf{x}_n, \mathsf{alg}\ \mathbf{z}_l, \mathsf{chan}\ \mathbf{h}_m, i, L_R\ \text{`}|\text{'}p\ ]\!]$$

is used to declare a scope consisting of local discrete variables $s_1, \ldots, s_k$, local (non-jumping) continuous variables $x_1, \ldots, x_n$, local algebraic variables $z_1, \ldots, z_l$, local channels $h_1, \ldots, h_m$, initialization predicate i, and local recursion definition list L_R. The variables all have to be different.

$$\begin{array}{l} [\![\ \mathsf{disc}\ \mathbf{s}_k \\ ,\ \mathsf{cont}\ \mathbf{x}_n \\ ,\ \mathsf{alg}\ \mathbf{z}_l \\ ,\ \mathsf{chan}\ \mathbf{h}_m \\ ,\ i \\ ,\ L_R \\ \mid\ p \\ ]\!] \end{array} \triangleq \begin{array}{l} [\![_{\mathrm{V}}\ \sigma_{sx} \\ \quad ,\ \{x_1, \ldots, x_n\} \\ \quad ,\ \{z_1, \ldots, z_l\} \\ \quad \mid\ [\![_{\mathrm{H}}\ \{h_1, \ldots, h_m\} \\ \qquad \mid\ \upsilon_{\{h_1,\ldots,h_m\}}([\![_{\mathrm{R}}\ \{L_R\} \mid i \curvearrowright p\ ]\!]) \\ \qquad]\!] \\ ]\!] \end{array}$$

Here L_R denotes the recursion definitions $X_1 \mapsto p_1, \ldots, X_r \mapsto p_r$, σ_{sx} denotes a valuation with $\mathrm{dom}(\sigma_{sx}) = \{s_1, \ldots, s_k,\ x_1, \ldots, x_n\}$, and σ_{sx} is undefined for all elements from its domain: $\forall_{v \in \mathrm{dom}(\sigma_{sx})}\, \sigma_{sx}(v) = \bot$.

In a similar way as defined for χ processes, the keyword preceding variables of a certain type is omitted when there are no variables of that type, and the keyword chan is omitted when there are no local channel declarations. Also the initialization predicate i and the recursion definitions may be omitted, indicating a predicate that always holds and an empty list of recursion definitions, respectively.

Process instantiation Process instantiation process term $l_{\mathrm{p}}(\mathbf{x}_k, \mathbf{h}_m, \mathbf{e}_n)$, where l_{p} denotes a process label, enables (re)-use of a process definition. A process definition is specified once, but it can be instantiated many times, possibly with different parameters: external variables $\mathbf{x}_k$, external channels $\mathbf{h}_m$, and expressions $\mathbf{e}_n$.

Chi specifications in which process instantiations $l_{\mathrm{p}}(\mathbf{x}_k, \mathbf{h}_m, \mathbf{e}_n)$ are used have the following structure:

$$
\begin{array}{l}
pd_1 \\
\vdots \\
pd_j \\
\\
\langle\ \mathsf{disc}\ \ldots,\ \mathsf{cont}\ \ldots,\ \mathsf{alg}\ \ldots,\ \mathsf{chan}\ \ldots,\ i,\ L_R \mid q\ \rangle,
\end{array}
$$

where for each process instantiation $l_{\mathrm{p}}(\mathbf{x}_k, \mathbf{h}_m, \mathbf{e}_n)$ occurring in process term q, a matching process definition pd_j of the form

$$l_{\mathrm{p}}(\mathsf{ext}\ \mathbf{x}'_k, \mathsf{chan}\ \mathbf{h}'_m, \mathsf{val}\ \mathbf{v}_n) = p$$

must be present among the j process definitions $pd_1 \ldots pd_j$. Here l_{p} denotes a process label, $\mathbf{x}_k$ denotes the 'actual external' variables $x_1, \ldots, x_k$, $\mathbf{h}_m$ denotes the 'actual external' channels $h_1, \ldots, h_m$, $\mathbf{e}_n$ denotes the expressions $e_1, \ldots, e_n$, $\mathbf{x}'_k$ denotes the 'formal external' variables $x'_1, \ldots, x'_k$, $\mathbf{h}'_m$ denotes the 'formal external' channels $h'_1, \ldots, h'_m$, and $\mathbf{v}_n$ denotes the 'value parameters' $v_1, \ldots, v_n$.

The only free variables and free channels that are allowed in process term p are the formal external variables $\mathbf{x}'_k$, the formal external channels $\mathbf{h}'_m$, and the value parameters $\mathbf{v}_n$. We assume that the formal external variables $\mathbf{x}'_k$ and the value parameters $\mathbf{v}_n$ are different.

Formally, the syntactic translation of process instantiation

$$l_{\mathrm{p}}(\mathbf{x}_k, \mathbf{h}_m, \mathbf{e}_n)$$

with corresponding process definition

$$l_{\mathrm{p}}(\mathsf{ext}\ \mathbf{x}'_k, \mathsf{chan}\ \mathbf{h}'_m, \mathsf{val}\ \mathbf{v}_n) = p$$

is given by

$$
\begin{array}{l}
[\![_{\mathrm{V}}\ \{v_1 \mapsto \bot, \ldots, v_n \mapsto \bot\},\ \emptyset,\ \emptyset \\
\mid\ \mathbf{v}_n = \mathbf{w}_n \curvearrowright p \\
]\!]\ [\mathbf{x}_k, \mathbf{h}_m, \mathbf{e}_n / \mathbf{x}'_k, \mathbf{h}'_m, \mathbf{w}_n].
\end{array}
$$

Notation $p[\mathbf{x}_k, \mathbf{h}_m, \mathbf{e}_n / \mathbf{x}'_k, \mathbf{h}'_m, \mathbf{w}_n]$ denotes the process term obtained from p by substitution of the (free) variables $\mathbf{x}'_k$ by $\mathbf{x}_k$, of the (free) channels $\mathbf{h}'_m$ by $\mathbf{h}_m$, and of the (free) variables $\mathbf{w}_n$ by expressions $\mathbf{e}_n$.

The variables $\mathbf{w}_n$ are assumed to be fresh with respect to $\mathbf{x}'_k$ and $\mathbf{v}_n$. The substitution is defined in such a way that no variables from $\mathbf{x}_k$ or $\mathbf{e}_n$, and no channels from $\mathbf{h}_m$ become bound. If substitution would cause new bindings,

the local variable or local channel that a variable or channel from $\mathbf{x}_k$, $\mathbf{e}_n$, or $\mathbf{h}_m$ would become bound to, is renamed into a fresh variable or fresh channel before the substitution takes place.

The translation declares the value parameters $\mathbf{v}_n$ as local discrete variables with initial values $\mathbf{e}_n$. By convention, however, process term p normally does not change the values of these variables.

2.6 Data types

The χ formalism is statically strongly typed. Besides the classification of variables as defined before, all variables have a type. The type of a variable defines the allowed values of the variable and the allowed operations on the variable. The atomic types are nat (natural numbers, including zero), int (integers), real (real-valued numbers), bool (booleans), string (strings), and enum (enumerations). Type constructors operate on existing types to create structured types. The χ formalism defines type constructors to create sets, lists, array tuples, record tuples, dictionaries, functions, and distributions (for stochastic models). Channels also have a type that indicates the type of data that is communicated via the channel. Pure synchronization channels, that do not communicate data, are of the predefined type void. The χ type system is strictly enforced in the χ tools. However, since the type system is not formalized, it is omitted from the specifications in this paper.

3 Semantics of the Chi formalism

This section presents the structured operational semantics (SOS [22]) of χ. It associates a hybrid transition system [23] with a χ process.

3.1 General description of the SOS

The main purpose of such an SOS is to define the behavior of χ processes at a certain chosen level of abstraction. The meaning of a χ process depends on the values of the variables and on the environment. A set $\mathcal{V}$ of variables, and a set $\mathcal{H}$ of channel labels are assumed. The values of the variables at a specific moment in time are captured by means of a valuation, i.e., a partial function from the variables to the set of values Λ (containing at least the booleans $\mathbb{B}$ and the reals $\mathbb{R}$). The set of all valuations is denoted Σ: $\Sigma = \mathcal{V} \mapsto \Lambda$, and we assume $\sigma \in \Sigma$ for all χ processes $\langle p, \sigma, E \rangle$. Extended valuations also include

the values of dotted continuous variables and the algebraic variables. The set of all extended valuations is denoted $\dot{\Sigma}$: $\dot{\Sigma} = (\mathcal{V} \cup \dot{\mathcal{V}}) \mapsto \Lambda$, where $\dot{\mathcal{V}}$ denotes the set of all dotted variables. The set T is used to represent points in time; usually $T = \mathbb{R}_{\geq 0}$. The set of environments $\mathcal{E}$ is defined as $\mathcal{E} = \mathcal{P}(\mathcal{V}) \times \mathcal{P}(\mathcal{V}) \times \mathcal{P}(\mathcal{V}) \times \mathcal{P}(\mathcal{H}) \times \mathcal{R}$, where $\mathcal{R} = \mathcal{X} \mapsto P$ denotes the set of all partial functions of recursion variables $\mathcal{X}$ to process terms P.

The SOS is chosen to represent the following:

(1) Discrete behavior by means of action transitions:
   (a) $\_ \xrightarrow{\cdot} \_ \subseteq (P \times \Sigma \times \mathcal{E}) \times (\dot{\Sigma} \times \mathcal{A} \times \dot{\Sigma}) \times (P \times \Sigma \times \mathcal{E})$, where $\mathcal{A}$ denotes the set of actions, and is defined as $\mathcal{A} = A_{\text{label}} \cup A_{\text{com}}$. The set of action labels A_{label} includes at least the pre-defined internal action τ. The set of communication actions A_{com} is defined as $A_{\text{com}} = \{\text{isa}(h, cs), \text{ira}(h, cs, W), \text{ca}(h, cs) \mid h \in \mathcal{H}, cs \in \Lambda^*, W \subseteq \mathcal{V}\}$, where isa, ira, and ca denote action labels for the internal send action, the internal receive action, and the communication action respectively, $h \in \mathcal{H}$ denotes a channel, $cs \in \Lambda^*$ denotes a list $[c_1, \ldots, c_n]$ of values, and W denotes a set of variables. The intuition of an action transition $\langle p, \sigma, E\rangle \xrightarrow{\xi, a, \xi'} \langle p', \sigma', E'\rangle$ is that the process $\langle p, \sigma, E\rangle$ executes the discrete action $a \in \mathcal{A}$ with extended valuations ξ and ξ' and thereby transforms into the process $\langle p', \sigma', E'\rangle$, where σ' and E' denote the accompanying valuation and environment of the process term p', respectively, after the discrete action a is executed.
   (b) $\_ \xrightarrow{\cdot} \langle \checkmark, \_, \_\rangle \subseteq (P \times \Sigma \times \mathcal{E}) \times (\dot{\Sigma} \times \mathcal{A} \times \dot{\Sigma}) \times (\Sigma \times \mathcal{E})$. The intuition of a (termination) transition $\langle p, \sigma, E\rangle \xrightarrow{\xi, a, \xi'} \langle \checkmark, \sigma', E'\rangle$ is that the process $\langle p, \sigma, E\rangle$ executes the discrete action a with extended valuations ξ and ξ' and thereby transforms into the terminated process $\langle \checkmark, \sigma', E'\rangle$.

(2) Continuous behavior by means of time transitions: $\_ \overset{\cdot}{\longmapsto} \_ \subseteq (P \times \Sigma \times \mathcal{E}) \times (T \times (T \mapsto \dot{\Sigma})) \times (P \times \Sigma \times \mathcal{E})$. The intuition of a time transition $\langle p, \sigma, E\rangle \overset{t, \rho}{\longmapsto} \langle p', \sigma', E'\rangle$ is that during the time transition, the extended valuation at each time-point $s \in [0, t]$ is given by $\rho(s)$. At the end-point t, the resulting process is $\langle p', \sigma', E'\rangle$.

(3) Consistency by means of a predicate: $\_ \overset{\cdot}{\rightsquigarrow} \subseteq (P \times \Sigma \times \mathcal{E}) \times \dot{\Sigma}$. The intuition of a consistency predicate $\langle p, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$ is that the process term p is consistent with the extended valuation ξ in environment E.

The following properties of the semantics can be found in Section 5:

- For all transitions, the domain of the valuation σ equals the domain of valuation σ', and environment E equals environment E'.
- For all action transitions $\langle p, \sigma, E\rangle \xrightarrow{\xi, a, \xi'} \langle \checkmark, \sigma', E'\rangle$ and $\langle p, \sigma, E\rangle \xrightarrow{\xi, a, \xi'} \langle p', \sigma', E'\rangle$: $\text{dom}(\sigma) \subseteq \text{dom}(\xi)$, $\text{dom}(\xi) = \text{dom}(\xi')$, extended valuation ξ restricted to $\text{dom}(\sigma)$ equals valuation σ, and extended ξ' restricted to $\text{dom}(\sigma')$

equals valuation σ'.

- For all time transitions $\langle p, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle p, \sigma', E'\rangle$: $\mathrm{dom}(\rho) = [0, t]$, and for all variables $x \in \mathrm{dom}(\sigma)$, the value in the resulting valuation $\sigma'(x)$ equals the value of the variable in the end-point of the trajectory $\rho(t)(x)$.
- For all consistency predicates $\langle p, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$: extended valuation ξ restricted to $\mathrm{dom}(\sigma)$ equals valuation σ.

The relations and predicates mentioned above are defined through so-called deduction rules. A deduction rule is of the form $\frac{H}{r}$, where H is a number of hypotheses separated by commas and r is the result of the rule. The result of a deduction rule can be derived if all of its hypotheses are derived. In case the set of hypotheses is empty, the deduction rule is called an axiom.

In order to increase the readability of the χ deduction rules, some additional abbreviations are used. Notation $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle q, \sigma'\rangle$, where $q \in P \cup \{\checkmark\}$ is an abbreviation for $\langle p, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle q, \sigma', E\rangle$, notation $E \Vdash \langle p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle q, \sigma'\rangle$ is an abbreviation for $\langle p, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q, \sigma', E\rangle$, and notation $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ is an abbreviation for $\langle p, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$.

Notation $E \Vdash f_1, \ldots, f_n$, where f_i represents one of the previously defined transition relations (of the forms $\langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle q, \sigma'\rangle$ or $\langle p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle q, \sigma'\rangle$ or $\langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$) is an abbreviation for $E \Vdash f_1, \ldots, E \Vdash f_n$.

Notation

$$\frac{E' \Vdash \langle p_1, \sigma_1\rangle \xrightarrow{\xi_1,a_1,\xi_1'} \left\langle \begin{matrix} q_{11} \\ \vdots \\ q_{1n} \end{matrix} , \sigma_1' \right\rangle, \ldots, \langle p_m, \sigma_m\rangle \xrightarrow{\xi_m,a_m,\xi_m'} \left\langle \begin{matrix} q_{m1} \\ \vdots \\ q_{mn} \end{matrix} , \sigma_m' \right\rangle,\ C}{E \Vdash \langle r, \sigma\rangle \xrightarrow{\xi,b,\xi'} \left\langle \begin{matrix} s_1 \\ \vdots \\ s_n \end{matrix} , \sigma' \right\rangle}$$

where $q_{j_i}, s_i \in P \cup \{\checkmark\}$, $p_i, r \in P$, and C denotes an optional hypothesis that must be satisfied in the deduction rule, is an abbreviation for the following rules (one for each i):

$$\frac{E' \Vdash \langle p_1, \sigma_1\rangle \xrightarrow{\xi_1,a_1,\xi_1'} \langle q_{1i}, \sigma_1'\rangle, \ldots, \langle p_m, \sigma_m\rangle \xrightarrow{\xi_m,a_m,\xi_m'} \langle q_{mi}, \sigma_m'\rangle,\ C}{E \Vdash \langle r, \sigma\rangle \xrightarrow{\xi,b,\xi'} \langle s_i, \sigma'\rangle}$$

The notation $\frac{H}{R}$, where R is a number of results separated by commas, is an

abbreviation for a set of deduction rules of the form $\frac{H}{r}$; one for each $r \in R$, and notation ${}_E\frac{H}{r}$ is an abbreviation for $\frac{E \Vdash H}{E \Vdash r}$.

Furthermore, notation $\langle p, \sigma, E\rangle \overset{\mathrm{ca}(h,*)}{\nrightarrow}$ denotes $(\nexists_{\xi,cs,\xi',p',\sigma',E'} \langle p, \sigma, E\rangle \xrightarrow{\xi,\mathrm{ca}(h,cs),\xi'} \langle p', \sigma', E'\rangle) \wedge (\nexists_{\xi,cs,\xi',\sigma',E'} \langle p, \sigma, E\rangle \xrightarrow{\xi,\mathrm{ca}(h,cs),\xi'} \langle \checkmark, \sigma', E'\rangle)$, and notation $\langle p, \sigma, E\rangle \xrightarrow{\alpha} \langle p', \sigma', E'\rangle$ is an abbreviation for $\langle p, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle p', \sigma', E'\rangle$ for some ξ, a, and ξ'.

3.2 Notations and mathematical definitions

Notations $f \in M \to G$ and $g \in M \mapsto G$ define complete function f, $\mathrm{dom}(f) = M$, and partial function g, $\mathrm{dom}(g) \subseteq M$, both with range G.

3.2.1 Operators on functions

Based on [10], the following definitions of operators $\restriction$, $\cup$, and $\downarrow$ applied on functions are used. If f is a function, $\mathrm{dom}(f)$ and $\mathrm{range}(f)$ denote the domain and range of f, respectively. If S is a set, $f \restriction S$ denotes the restriction of f to S, that is, the function g with $\mathrm{dom}(g) = \mathrm{dom}(f) \cap S$, such that $g(c) = f(c)$ for each $c \in \mathrm{dom}(g)$.

If f and g are functions with $\mathrm{dom}(f) \cap \mathrm{dom}(g) = \emptyset$, then $f \cup g$ denotes the unique function h with $\mathrm{dom}(h) = \mathrm{dom}(f) \cup \mathrm{dom}(g)$ satisfying the condition: for each $c \in \mathrm{dom}(h)$, if $c \in \mathrm{dom}(f)$ then $h(c) = f(c)$, and $h(c) = g(c)$ otherwise.

If f is a function whose range is a set of functions and S is a set, then $f \downarrow S$ denotes the function g with $\mathrm{dom}(g) = \mathrm{dom}(f)$ such that $g(c) = f(c) \restriction S$ for each $c \in \mathrm{dom}(g)$. If f is a function whose range is a set of functions, all of which have a particular element d in their domain, then $f \downarrow d$ denotes the function g with $\mathrm{dom}(g) = \mathrm{dom}(f)$ such that $g(c) = f(c)(d)$ for each $c \in \mathrm{dom}(g)$.

3.2.2 Notations

Let $x \in \mathcal{V}$ be a variable, $S, C, L \subseteq \mathcal{V}$ be sets of variables, $\sigma \in \Sigma$ be a valuation, e be an expression over variables and constants, and $t \in T$ be a time-point, then the following notations are defined:

- $\sigma(x)$ denotes the value of variable x in valuation σ. We use the similar notation $\sigma(e)$.
- $\dot{S}$ denotes the set of dotted variables $\{\dot{x} \mid x \in S\}$.
- $\xi^{\dot{C}L} \in (\dot{C} \cup L) \to \Lambda$ denotes an arbitrary valuation with domain $\dot{C} \cup L$.

- ξ_σ is an abbreviation for $\xi \restriction \mathrm{dom}(\sigma)$.
- Function $\Xi \in (\Sigma \times \mathcal{P}(\mathcal{V}) \times \mathcal{P}(\mathcal{V}) \times \mathcal{P}(\mathcal{V})) \to \mathcal{P}(\dot{\Sigma})$ returns a set of extended valuations, given a valuation, the set of continuous variables, the set of jumping variables, and the set of algebraic variables. Formally, function Ξ is defined as:

  $$\Xi(\sigma, C, J, L) = \{\xi \mid \mathrm{dom}(\xi) = \mathrm{dom}(\sigma) \cup \dot{C} \cup L,\ \forall_{x \in \mathrm{dom}(\sigma) \setminus J}\, \xi(x) = \sigma(x)\}.$$

  The values of the variables in ξ are defined as follows: the values of the variables in $\mathrm{dom}(\sigma) \setminus J$ are given by σ. The jumping variables J, the dotted variables $\dot{C}$ and the algebraic variables L have arbitrary values.
- $\Omega_{\sigma E t}$, where environment E denotes the tuple (C, J, L, H, R), is an abbreviation for $\Omega_{FG}(\sigma, C, L, \mathrm{true}, t)$. Here, Ω_{FG} is the solution function as defined in Section 3.3.2.
- ρ_σ is an abbreviation for $\rho \downarrow \mathrm{dom}(\sigma)$.

3.3 Deduction rules for atomic process terms

3.3.1 Action predicate

Action predicate process term $W : r \gg l_\mathrm{a}$ denotes instantaneous changes to the variables from set W, by means of an action labeled $l_\mathrm{a} \in A_\mathrm{label}$, such that predicate r over variables from the domains of the extended valuations ξ^- and ξ' is satisfied, see Rule 1, where $\xi, \xi' \in (\mathrm{dom}(\sigma) \cup \dot{C} \cup L) \to \Lambda$, and ξ^- is defined below.

The values of the variables from $\mathrm{dom}(\sigma)$ in ξ are given by σ. The dotted variables $\dot{C}$ and the algebraic variables L in ξ can in principle take any value ($\xi = \sigma \cup \xi^{\dot{C}L}$) as long as the action predicate r is satisfied ($\xi^- \cup \xi' \models r$). Variables occurring with a '$^-$' superscript in r are evaluated in ξ^-, which denotes an extended valuation with $\mathrm{dom}(\xi^-) = \{x^- \mid x \in \mathrm{dom}(\xi)\}$, and $\xi^-(x^-) = \xi(x)$. For extended valuation ξ', the values of the discrete and the non-jumping variables ($\mathrm{dom}(\sigma) \setminus (J \cup W)$) are given by σ. The jumping variables J, the variables from set W, the dotted variables $\dot{C}$ and the algebraic variables L are allowed to change such that the action predicate is satisfied.

Rule 2 states that action predicates are consistent with any extended valuation $\sigma \cup \xi^{\dot{C}L}$ in any environment E.

$$\frac{\xi = \sigma \cup \xi^{\dot{C}L},\ \xi' \in \Xi(\sigma, C, J \cup W, L),\ \xi^- \cup \xi' \models r}{(C, J, L, H, R) \Vdash \langle W : r \gg l_\mathrm{a}, \sigma \rangle \xrightarrow{\xi,\ l_\mathrm{a},\ \xi'} \langle \checkmark, \xi'_\sigma \rangle}\ 1$$

$$\frac{}{(C, J, L, H, R) \Vdash \langle W : r \gg l_\mathrm{a}, \sigma \rangle \overset{\sigma \cup \xi^{\dot{C}L}}{\rightsquigarrow}}\ 2$$

3.3.2 Delay predicate

Delay predicate u is a predicate over variables and dotted continuous variables.

$$\frac{\rho \in \Omega_{FG}(\sigma, C, L, u, t)}{(C, J, L, H, R) \Vdash \langle u, \sigma \rangle \overset{t,\rho}{\longmapsto} \langle u, \rho_\sigma(t) \rangle}\ 3 \qquad \frac{\sigma \cup \xi^{\dot{C}L} \models u}{(C, J, L, H, R) \Vdash \langle u, \sigma \rangle \overset{\sigma \cup \xi^{\dot{C}L}}{\rightsquigarrow}}\ 4$$

Function $\Omega_{FG} \in \Sigma \times \mathcal{P}(\mathcal{V}) \times \mathcal{P}(\mathcal{V}) \times U \times T \to \mathcal{P}(T \mapsto \dot{\Sigma})$, where U denotes the set of all predicates over $\mathcal{V}$ and $\dot{\mathcal{V}}$, returns a set of trajectories from time to an extended valuation for the variables and dotted variables, given a valuation representing the current values of the discrete and continuous variables, the set of continuous variables, the set of algebraic variables, a delay predicate and a time point that denotes the duration of the trajectory. Formally, function Ω_{FG} is defined as:

$$\begin{array}{ll}
\Omega_{FG}(\sigma, C, L, u, t) = & \\
\{\, \rho & \\
\mid\ \rho \in [0, t] \to ((\mathrm{dom}(\sigma) \cup \dot{C} \cup L) \to \Lambda) & \\
,\ t \geq 0 & \\
,\ \forall s \in [0, t] : & \rho(s) \models u \\
,\ \forall x \in \mathrm{dom}(\sigma) \setminus (\{\mathsf{time}\} \cup C) : & \rho \downarrow x \text{ is a constant function.} \\
,\ \forall x \in \mathrm{dom}(\sigma) : & (\rho \downarrow x)(0) = \sigma(x) \\
,\ \forall x \in L : & \rho \downarrow x \in F \\
,\ \forall x \in C : & \rho \downarrow \dot{x} \text{ is an integrable function in the} \\
 & \text{Lesbesgue sense.} \\
,\ \forall s \in [0, t], x \in C : & (\rho \downarrow x)(s) = (\rho \downarrow x)(0) + \int_0^s (\rho \downarrow \dot{x})(s')ds' \\
,\ \forall x \in C : & (\rho \downarrow x, \rho \downarrow \dot{x}) \in G \\
,\ \forall s \in [0, t] : & \rho(s)(\mathsf{time}) = \sigma(\mathsf{time}) + s \\
\} &
\end{array}$$

The trajectory ρ is a function from the time interval $[0, t]$, where $t \geq 0$, to a valuation, where the domain of each valuation consists of all variables and dotted continuous variables. Trajectory ρ satisfies the predicate u for all time points of its domain ($\forall_{s \in [0,t]}\, \rho(s) \models u$). The trajectory of each discrete variable $x \in \mathrm{dom}(\sigma) \setminus (\{\mathsf{time}\} \cup C)$ is restricted to a constant function. The initial value (starting-point) of the trajectory of each discrete and continuous variable equals the value of that variable in σ ($\forall_{x \in \mathrm{dom}(\sigma)}\, (\rho \downarrow x)(0) = \sigma(x)$).

The trajectories of the algebraic variables ($\rho \downarrow x$ for $x \in L$) are required to be functions of type F. This set of functions is a parameter of the solution concept

of χ. The definition of the trajectory as $\rho \in [0,t] \to ((\mathrm{dom}(\sigma) \cup \dot{C} \cup L) \to \Lambda)$ ensures that $\forall_{x \in L}\, (\rho \downarrow x) \in [0,t] \to \Lambda$. Having the set F as a parameter of the solution concept allows us to restrict F to, for instance, the set of piecewise constant functions, if this would be required for certain properties to hold.

The trajectories of the dotted variables are required to be integrable. This ensures that the integral $\int_0^s (\rho \downarrow \dot{x})(s')ds'$ is defined. The relation between the trajectory of a continuous variable x and the trajectory of its 'derivative' $\dot{x}$ is given by the Caratheodory solution concept [3]: $(\rho \downarrow x)(s) = (\rho \downarrow x)(0) + \int_0^s (\rho \downarrow \dot{x})(s')ds'$. Note that this integral relation can hold only for those continuous variables for which $\rho \downarrow x$ is an absolutely continuous function. Thus the solution function Ω_{FG} restricts the trajectory $\rho \downarrow x$ of every continuous variable x to an absolutely continuous function, but it does allow a non-smooth trajectory for a continuous variable in the case that the trajectory of its 'derivative' $\rho \downarrow \dot{x}$ is non-smooth or even discontinuous, as in, for example, $\langle\, \mathsf{cont}\ y,\ y = 0 \mid \dot{y} = \mathrm{step}(\mathsf{time} - 1)\, \rangle$, where $\mathrm{step}(x)$ equals 0 for $x \leq 0$ and 1 for $x > 0$.

The disadvantage of the Caratheodory solution concept is that it introduces spurious solutions in a higher index system such as $\langle\, \mathsf{cont}\, y,\ \mathsf{alg}\, z \mid y = \mathsf{time}, z = \dot{y}\, \rangle$. Here, one could argue that the trajectory for z should be the constant function 1. The Caratheodory solution concept, however, allows trajectories for z that are 1, except for discontinuity points, where any other value is allowed. Such spurious discontinuities in $\rho \downarrow \dot{x}$, in the case that the trajectory of a differential variable x is smooth, and thus $\rho \downarrow x$ is differentiable, on some interval I, can be prevented in two ways.

First, by changing the model to $\langle\, \mathsf{cont}\ y, z \mid y = \mathsf{time}, z = \dot{y}\, \rangle$. Defining z as a continuous variable requires its trajectory to be (absolutely) continuous.

Second, by restricting the solution concept. This can be done by restricting set G in the requirement $\forall_{x \in C}\, (\rho \downarrow x, \rho \downarrow \dot{x}) \in G$, where G is a parameter of the χ solution concept. Defining $G = \{(f, f') \mid \forall_{I \subseteq \mathrm{dom}(f)}\, f$ is differentiable on $I \Rightarrow f' \restriction I$ is the derivative function of $f \restriction I\}$, where I denotes some interval, requires the solution function $\rho \downarrow \dot{x}$ for the dotted variable $\dot{x}$ to be indeed the derivative function of the solution function $\rho \downarrow x$ for the differential variable x, for all intervals where $\rho \downarrow x$ is differentiable. This prevents spurious discontinuities from occurring in higher index systems as discussed above. The disadvantage of this set G is that for instance the delay predicate $(\mathsf{time} = 1 \Rightarrow \dot{x} = 1) \wedge (\mathsf{time} \neq 1 \Rightarrow \dot{x} = 0)$ has no solution for x $(\rho \downarrow x)$ on the interval $[0,t]$, for $t > 1$, starting from a valuation in which $\mathsf{time} = 0$. A constant function of time for x with domain $[0,t]$ for $t > 0$, which is a solution for $G = \{(f, f') \mid \mathrm{true}\}$, is not a solution for the restricted version of G defined above, because the derivative function (here $\rho \downarrow \dot{x}$) of a constant function (here $\rho \downarrow x$) is always zero, and therefore the valuation at time point 1 ($\rho(1)$) does not satisfy the delay predicate.

The properties derived in Section 5.2 are valid for all parameters F and G. For the translation of a hybrid automaton to χ as defined in [2], differentiable functions are assumed for the trajectories of the continuous variables: $G = \{(f, f') \mid f \text{ is differentiable, and } f' \text{ is the derivative function of } f\}$. In this way, the semantics of the χ translation corresponds to the semantics of the hybrid automaton. For the examples in Section 4, differentiability would be too strong a restriction. Therefore, piecewise continuous functions for the trajectories of the algebraic and dotted variables are assumed: $F = \{f \mid f \text{ is a piecewise continuous function}\}$, $G = \{(f, f') \mid f' \text{ is a piecewise continuous function}\}$. There is no fundamental reason for this choice. Another possibility would have been not to define additional restrictions: $F = \{f \mid \text{true}\}$, $G = \{(f, f') \mid \text{true}\}$. For a model with just one solution such as: $\langle\, \mathsf{cont}\ x,\ \mathsf{alg}\ y \mid \dot{x} = y,\ y = \text{step}(\mathsf{time} - 1)\,\rangle$, the solution is the same for both cases of F and G. For a model that allows infinitely many solutions, such as $\langle\, \mathsf{cont}\ x,\ \mathsf{alg}\ y \mid \text{true}\,\rangle$, there would obviously be a difference.

In some deduction rules describing delay behavior, abbreviation $\Omega_{\sigma Et}$, which denotes $\Omega_{FG}(\sigma, C, L, \text{true}, t)$, is used as a hypothesis. The true predicate does not restrict t and the trajectory ρ other than by means of the default restrictions. Among others, the discrete variables remain constant, and the trajectory of each continuous variable is an absolutely continuous function that starts with the value of the continuous variable in σ.

3.3.3 Send and receive

Send and receive process terms $h\,!!\,\mathbf{e}_n$ and $h\,??\,\mathbf{x}_n$ denote undelayable sending of expression $\mathbf{e}_n$ via channel h, and undelayable receiving of information via channel h into variable(s) $\mathbf{x}_n$, respectively.

The values of expressions $e_1, \ldots, e_n$ which are sent via channel h are evaluated in extended valuation ξ, see Rule 5, where $\mathbf{e}_n$ denotes $e_1, \ldots, e_n$, $[\xi(\mathbf{e}_n)]$ denotes the list of values $[\xi(e_1), \ldots, \xi(e_n)]$ for $n \geq 1$, and $\xi(e)$ denotes the value of expression e for extended valuation ξ. The case that n equals 0, represents the case where nothing is sent via the channel, and $\mathbf{e}_0$ and $[\xi(\mathbf{e}_0)]$ denote an empty expression and an empty list, respectively. For $n \geq 1$, the receive process term $h\,??\,x_1, \ldots, x_n$ can receive the list of values $[c_1, \ldots, c_n]$, see Rule 6, where $\mathbf{x}_n$ denotes $x_1, \ldots, x_n$, $\{\mathbf{x}_n\}$ denotes the set $\{x_1, \ldots, x_n\}$, $[\mathbf{c}_n]$ denotes the list of values $[c_1, \ldots, c_n]$, and $\xi'(\mathbf{x}_n) = \mathbf{c}_n$ is an abbreviation for $\xi'(x_1) = c_1, \ldots, \xi'(x_n) = c_n$. For $n = 0$, nothing is received, so that $\mathbf{x}_0$ and $\mathbf{c}_0$ are empty, and $\xi'(\mathbf{x}_0) = \mathbf{c}_0$ always holds.

$$\frac{\xi = \sigma \cup \xi^{\dot{C}L},\ \xi' \in \Xi(\sigma, C, J, L)}{(C, J, L, H, R) \Vdash \langle h\,!!\,\mathbf{e}_n, \sigma\rangle \xrightarrow{\xi,\ \text{isa}(h, [\xi(\mathbf{e}_n)]),\ \xi'} \langle \checkmark, \xi'_\sigma\rangle}\ 5$$

$$\frac{\xi = \sigma \cup \xi^{\dot{C}L},\ \xi' \in \Xi(\sigma, C, J \cup \{\mathbf{x}_n\}, L),\ \xi'(\mathbf{x}_n) = \mathbf{c}_n}{(C, J, L, H, R) \Vdash \langle h \,??\, \mathbf{x}_n, \sigma\rangle \xrightarrow{\xi,\ \mathrm{ira}(h,[\mathbf{c}_n],\{\mathbf{x}_n\}),\ \xi'} \langle \checkmark, \xi'_\sigma\rangle}\ 6$$

$$\frac{}{(C, J, L, H, R) \Vdash \langle h \,!!\, \mathbf{e}_n, \sigma\rangle \overset{\sigma\cup\xi^{\dot{C}L}}{\rightsquigarrow}}\ 7 \qquad \frac{}{(C, J, L, H, R) \Vdash \langle h \,??\, \mathbf{x}_n, \sigma\rangle \overset{\sigma\cup\xi^{\dot{C}L}}{\rightsquigarrow}}\ 8$$

3.3.4 Deadlock and inconsistent process term

Process term δ cannot perform any action transitions, nor time transitions. It is, however, consistent for arbitrary extended valuations $\sigma \cup \xi^{\dot{C}L}$.

$$\frac{}{(C, J, L, H, R) \Vdash \langle \delta, \sigma\rangle \overset{\sigma\cup\xi^{\dot{C}L}}{\rightsquigarrow}}\ 9$$

There are no rules for the inconsistent process term $\bot$. Therefore, it cannot do actions transition, nor time transitions, and it is inconsistent for all valuations and environments. Process term $\bot$ originates from the process algebra with propositional signals $\mathrm{ACP}_{\mathrm{ps}}$ ([24]).

3.4 Deduction rules for operators

3.4.1 Any delay operator

The any delay operator $[p]$ allows arbitrary time transitions, that need to satisfy only the general solution function requirements (e.g. trajectories of discrete variables are constant), regardless of the time transitions allowed by p (see Rule 11). The any delay operator does not affect the action behavior of p (see Rule 10). Process term $[p]$ is consistent with any extended valuation $\sigma \cup \xi^{\dot{C}L}$, in any environment E (see Rule 12).

$$E\ \frac{\langle p, \sigma\rangle \xrightarrow{\alpha} \langle \overset{\checkmark}{p'}, \sigma'\rangle}{\langle [p], \sigma\rangle \xrightarrow{\alpha} \langle \overset{\checkmark}{p'}, \sigma'\rangle}\ 10 \qquad E\ \frac{\rho \in \Omega_{\sigma E t}}{\langle [p], \sigma\rangle \overset{t,\rho}{\longmapsto} \langle [p], \rho_\sigma(t)\rangle}\ 11$$

$$\frac{}{(C, J, L, H, R) \Vdash \langle [p], \sigma\rangle \overset{\sigma\cup\xi^{\dot{C}L}}{\rightsquigarrow}}\ 12$$

3.4.2 Signal emission operator

The signal emission operator $u \curvearrowright p$ ensures that p starts its behavior from an extended valuation ξ in which initialization predicate u is satisfied. This operator was inspired by the signal emission operator from the process algebra with propositional signals $\mathrm{ACP}_{\mathrm{ps}}$ [24], which was also used in [25].

$$E\ \frac{\langle p,\sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \overset{\checkmark}{p'},\sigma'\rangle,\ \xi \models u}{\langle u \curvearrowright p,\sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \overset{\checkmark}{p'},\sigma'\rangle}\ 13 \qquad E\ \frac{\langle p,\sigma\rangle \overset{t,\rho}{\longmapsto} \langle p',\sigma'\rangle,\ \rho(0) \models u}{\langle\ u \curvearrowright p,\sigma\rangle \overset{t,\rho}{\longmapsto} \langle p',\sigma'\rangle}\ 14$$

$$E\ \frac{\langle p,\sigma\rangle \overset{\xi}{\rightsquigarrow},\ \xi \models u}{\langle\ u \curvearrowright p,\sigma\rangle \overset{\xi}{\rightsquigarrow}}\ 15$$

3.4.3 Sequential composition operator

The sequential composition of process terms p and q behaves as process term p until p terminates, and then continues to behave as process term q. When p terminates, its right-hand extended valuation ξ' must be consistent with q (see Rule 16).

$$E\ \frac{\langle p,\sigma\rangle \xrightarrow{\xi,a,\xi'} \langle\checkmark,\sigma'\rangle,\ \langle q,\sigma'\rangle \overset{\xi'}{\rightsquigarrow}}{\langle p;\ q,\sigma\rangle \xrightarrow{\xi,a,\xi'} \langle q,\sigma'\rangle}\ 16 \qquad E\ \frac{\langle p,\sigma\rangle \xrightarrow{\alpha} \langle p',\sigma'\rangle}{\langle p;\ q,\sigma\rangle \xrightarrow{\alpha} \langle p';\ q,\sigma'\rangle}\ 17$$

$$E\ \frac{\langle p,\sigma\rangle \overset{t,\rho}{\longmapsto} \langle p',\sigma'\rangle}{\langle p;\ q,\sigma\rangle \overset{t,\rho}{\longmapsto} \langle p';\ q,\sigma'\rangle}\ 18 \qquad E\ \frac{\langle p,\sigma\rangle \overset{\xi}{\rightsquigarrow}}{\langle p;\ q,\sigma\rangle \overset{\xi}{\rightsquigarrow}}\ 19$$

3.4.4 Guard operator

The guarded process term $b \to p$ can do whatever actions p can do under the condition that the guard evaluates to true using extended valuation ξ. Evaluating the guard in ξ ensures that when guard operators are nested with signal emission operators, actions can be executed only if all predicates of the signal emission operators and all guards hold, independently of the order. Furthermore, the values of the dotted variables and algebraic variables are defined in ξ, whereas they are not defined in σ.

The guarded process term can delay according to p under the condition that for all intermediate valuations the guard evaluates to true ($\forall_{s\in[0,t]}\, \rho(s) \models b$,

see Rule 21).

The guarded process term can perform arbitrary delays under the condition that for the intermediate valuations, possibly excluding the first and last valuation, the guard does not hold ($\forall_{s\in(0,t)}\, \rho(s) \models \neg b$). This ensures that, for example, the process $\langle\ \mathsf{disc}\ x,\ x = 1 \mid \mathsf{time} \geq x \rightarrow \mathrm{skip}\ \rangle$ behaves as expected: it can first do a time transition of 1, such that the value of the current time time becomes 1, and thereafter it can do a τ action to the terminated process. If the condition in Rule 22 would be $\forall_{s\in[0,t]}\, \rho(s) \models \neg b$, then a time transition of 1 would be impossible. This is because the value of the guard should then also be false for the last time point of the time transition, so that the point where the value of time equals 1 could not be reached. The condition $\rho(0) \models b \Rightarrow \langle p, \sigma\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle p', \sigma'\rangle$ in Rule 22, which states that p must be able to delay for a duration of 0 if the guard is initially true, ensures that undelayable actions in p have priority over delay behavior of a guard that is initially true and continues as false. The condition $\rho(t) \models b \Rightarrow \langle p, \rho_\sigma(t)\rangle \overset{\rho(t)}{\rightsquigarrow}$ in Rule 22 requires consistency if the guard holds in the end-point of the trajectory. This ensures that it is impossible to delay to an inconsistent state.

Finally, $b \rightarrow p$ is consistent with extended valuations for which b holds and with which p is consistent (Rule 23), and with extended valuations for which b does not hold (Rule 24).

$$E\ \frac{\langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \begin{smallmatrix}\checkmark\\ p'\end{smallmatrix}, \sigma'\rangle,\ \xi \models b}{\langle b \rightarrow p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \begin{smallmatrix}\checkmark\\ p'\end{smallmatrix}, \sigma'\rangle}\ 20 \qquad E\ \frac{\langle p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle p', \sigma'\rangle,\ \forall_{s\in[0,t]}\, \rho(s) \models b}{\langle b \rightarrow p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle b \rightarrow p', \sigma'\rangle}\ 21$$

$$E\ \frac{\begin{array}{c}\rho \in \Omega_{\sigma Et},\ \forall_{s\in(0,t)}\, \rho(s) \models \neg b,\\ \rho(0) \models b \Rightarrow \langle p, \sigma\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle p', \sigma'\rangle,\\ \rho(t) \models b \Rightarrow \langle p, \rho_\sigma(t)\rangle \overset{\rho(t)}{\rightsquigarrow}\end{array}}{\langle b \rightarrow p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle b \rightarrow p, \rho_\sigma(t)\rangle}\ 22$$

$$E\ \frac{\langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow},\ \xi \models b}{\langle b \rightarrow p, \sigma\rangle \overset{\xi}{\rightsquigarrow}}\ 23 \qquad \frac{\sigma \cup \xi^{\dot{C}L} \models \neg b}{(C, J, L, H, R) \Vdash \langle b \rightarrow p, \sigma\rangle \overset{\sigma\cup\xi^{\dot{C}L}}{\rightsquigarrow}}\ 24$$

3.4.5 Alternative composition operator

Applying the alternative composition operator to process terms p and q models a non-deterministic choice between p and q for action transitions. Process

term p can perform action transitions only if the initial extended valuation is consistent with q, as specified in Rule 25. Consider for example the following process term: $y = 1 \between x := y$. This corresponds to a hybrid automaton with one location with flow predicate true, invariant $y = 1$, and an urgent outgoing edge with jump condition $x := y$. The invariant $y = 1$ ensures that the value of y equals 1 when the outgoing edge is taken.

The passage of time cannot result in making a choice between p and q, since the time transitions of the process terms p and q have to synchronize to obtain the time transition (with the same time step t and trajectory ρ) of their alternative composition as defined by Rule 26.

$$E \ \frac{\langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \begin{smallmatrix}\checkmark\\ p'\end{smallmatrix}, \sigma'\rangle,\ \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}}{\langle p \between q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \begin{smallmatrix}\checkmark\\ p'\end{smallmatrix}, \sigma'\rangle,\ \langle q \between p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \begin{smallmatrix}\checkmark\\ p'\end{smallmatrix}, \sigma'\rangle}\ 25$$

$$E \ \frac{\langle p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle p', \sigma'\rangle,\ \langle q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle q', \sigma'\rangle}{\langle p \between q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle p' \between q', \sigma'\rangle}\ 26 \qquad E \ \frac{\langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow},\ \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}}{\langle p \between q, \sigma\rangle \overset{\xi}{\rightsquigarrow}}\ 27$$

3.4.6 Parallel composition operator

The parallel composition of process terms p and q has as its behavior with respect to action transitions the interleaving of the behaviors of p and q (see Rule 29). Process term p can only perform action transitions from an extended valuation ξ which is consistent with q. Furthermore, the resulting extended valuation ξ' must be consistent with q (see Rule 29).

The parallel composition allows the synchronization of matching send and receive actions. A send action $\mathrm{isa}(h, cs)$ and a receive action $\mathrm{ira}(h', cs', W)$ match iff $h = h'$ and $cs = cs'$; i.e. the channels used for sending and receiving are the same, and also the values sent and the values received are identical. Furthermore, the resulting extended valuations ξ' of both the send action and the receive action have to be the same. In order to be able to receive values in variables of the same scope as the send process term, the variables of which the value changes due to the receive action are passed on to the send process term. This is achieved by means of set W on the receive action, and the addition of this set W to the set of jumping variables in the environment where the send action takes place (see Rule 28). The result of the synchronization is a communication action that is represented by $\mathrm{ca}(h, cs)$ as defined by Rule 28.

The time transitions of the process terms that are put in parallel have to

synchronize in the same way as for alternative composition, see Rules 26 and 30.

$$\frac{\begin{array}{c}(C, J \cup W, L, H, R) \Vdash \langle p, \sigma \rangle \xrightarrow{\xi, \mathrm{isa}(h,cs), \xi'} \left\langle \begin{array}{c} \checkmark \\ p' \\ \checkmark \end{array}, \sigma' \right\rangle, \\ (C, J, L, H, R) \Vdash \langle q, \sigma \rangle \xrightarrow{\xi, \mathrm{ira}(h,cs,W), \xi'} \left\langle \begin{array}{c} p' \\ \checkmark \\ q' \end{array}, \sigma' \right\rangle \end{array}}{(C, J, L, H, R) \Vdash \langle p \parallel q, \sigma \rangle \xrightarrow{\xi, \mathrm{ca}(h,cs), \xi'} \left\langle \begin{array}{c} q' \\ \checkmark \\ p' \parallel q' \end{array}, \sigma' \right\rangle, \quad \langle q \parallel p, \sigma \rangle \xrightarrow{\xi, \mathrm{ca}(h,cs), \xi'} \left\langle \begin{array}{c} p' \\ \checkmark \\ q' \parallel p' \end{array}, \sigma' \right\rangle} 28$$

$$E \ \frac{\langle q, \sigma \rangle \overset{\xi}{\rightsquigarrow}, \ \langle p, \sigma \rangle \xrightarrow{\xi, a, \xi'} \langle \begin{array}{c} \checkmark \\ p' \end{array}, \sigma' \rangle, \ \langle q, \sigma' \rangle \overset{\xi'}{\rightsquigarrow}}{\langle p \parallel q, \sigma \rangle \xrightarrow{\xi, a, \xi'} \langle \begin{array}{c} q \\ p' \parallel q \end{array}, \sigma' \rangle, \ \langle q \parallel p, \sigma \rangle \xrightarrow{\xi, a, \xi'} \langle \begin{array}{c} q \\ q \parallel p' \end{array}, \sigma' \rangle} 29$$

$$E \ \frac{\langle p, \sigma \rangle \overset{t,\rho}{\longmapsto} \langle p', \sigma' \rangle, \ \langle q, \sigma \rangle \overset{t,\rho}{\longmapsto} \langle q', \sigma' \rangle}{\langle p \parallel q, \sigma \rangle \overset{t,\rho}{\longmapsto} \langle p' \parallel q', \sigma' \rangle} 30$$

$$E \ \frac{\langle p, \sigma \rangle \overset{\xi}{\rightsquigarrow}, \ \langle q, \sigma \rangle \overset{\xi}{\rightsquigarrow}}{\langle p \parallel q, \sigma \rangle \overset{\xi}{\rightsquigarrow}} 31$$

3.4.7 Action encapsulation operator

The behavior of the action encapsulation applied to a process term $\partial_A(p)$ is the same as the behavior of its argument with the restriction that actions from the set A ($A \subseteq \mathcal{A} \setminus \{\tau\}$) cannot be executed (see Rule 32). Action encapsulation has no effect on time transitions and consistency, as defined by Rules 33 and 34.

$$E\ \frac{\langle p,\sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \begin{matrix}\checkmark\\ p'\end{matrix}, \sigma'\rangle,\ a \notin A}{\langle \partial_A(p),\sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \begin{matrix}\checkmark\\ \partial_A(p')\end{matrix}, \sigma'\rangle}\ 32$$

$$E\ \frac{\langle p,\sigma\rangle \overset{t,\rho}{\longmapsto} \langle p',\sigma'\rangle}{\langle \partial_A(p),\sigma\rangle \overset{t,\rho}{\longmapsto} \langle \partial_A(p'),\sigma'\rangle}\ 33 \qquad E\ \frac{\langle p,\sigma\rangle \overset{\xi}{\rightsquigarrow}}{\langle \partial_A(p),\sigma\rangle \overset{\xi}{\rightsquigarrow}}\ 34$$

3.4.8 Urgent communication operator

The urgent communication operator $\upsilon_H(p)$ gives communication actions via channels from set $H \subseteq \mathcal{H}$ a higher priority than time transitions. Action behavior and consistency are not affected by the urgent communication operator, see Rules 35 and 36. Time transitions are allowed only if at each intermediate state while delaying no communication actions via channels from H are possible.

$$E\ \frac{\langle p,\sigma\rangle \xrightarrow{\alpha} \langle \begin{matrix}\checkmark\\ p'\end{matrix}, \sigma'\rangle}{\langle \upsilon_H(p),\sigma\rangle \xrightarrow{\alpha} \langle \begin{matrix}\checkmark\\ \upsilon_H(p')\end{matrix}, \sigma'\rangle}\ 35 \qquad E\ \frac{\langle p,\sigma\rangle \overset{\xi}{\rightsquigarrow}}{\langle \upsilon_H(p),\sigma\rangle \overset{\xi}{\rightsquigarrow}}\ 36$$

$$E\ \frac{\begin{array}{l}\langle p,\sigma\rangle \overset{t,\rho}{\longmapsto} \langle p',\sigma'\rangle,\ \forall_{s\in[0,t)}\ (\langle p,\sigma\rangle \overset{s,\rho\restriction[0,s]}{\longmapsto} \langle p_s,\sigma_s\rangle,\\ \qquad\qquad \langle p_s,\sigma_s\rangle \overset{t-s,\rho_{-s}}{\longmapsto} \langle p',\sigma'\rangle,\\ \qquad\qquad \forall_{h\in\mathcal{H}}\ \langle p_s,\sigma_s,E\rangle \overset{\mathrm{ca}(h,*)}{\nrightarrow})\end{array}}{\langle \upsilon_H(p),\sigma\rangle \overset{t,\rho}{\longmapsto} \langle \upsilon_H(p'),\sigma'\rangle}\ 37$$

where ρ_{-s} denotes the trajectory ρ shifted left by s time-units and starting at 0: $\mathrm{dom}(\rho_{-s}) = [0, t-s]$, assuming $\mathrm{dom}(\rho) = [0,t]$, and $\forall_{t'\in\mathrm{dom}(\rho_{-s})}\ \rho_{-s}(t') = \rho(t'+s)$.

3.4.9 Recursion variable

A recursion variable process term X behaves as the process term given by $R(X)$. Here $R(X)$ is the process term that is defined for recursion variable X in function R. This is equivalent to syntactically replacing recursion variable X by its defining process term $R(X)$. Function R can be defined in the environment of the χ process directly, or by means of the recursion scope operator, see Section 3.4.13.

$$(C, J, L, H, R)\ \frac{\langle R(X), \sigma\rangle \xrightarrow{\alpha} \langle \begin{matrix}\checkmark\\ p'\end{matrix}, \sigma'\rangle}{\langle X, \sigma\rangle \xrightarrow{\alpha} \langle \begin{matrix}\checkmark\\ p'\end{matrix}, \sigma'\rangle}\ 38$$

$$(C, J, L, H, R)\ \frac{\langle R(X), \sigma\rangle \overset{t,\rho}{\longmapsto} \langle p', \sigma'\rangle}{\langle X, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle p', \sigma'\rangle}\ 39$$

$$(C, J, L, H, R)\ \frac{\langle R(X), \sigma\rangle \overset{\xi}{\rightsquigarrow}}{\langle X, \sigma\rangle \overset{\xi}{\rightsquigarrow}}\ 40$$

3.4.10 Jump enabling operator

The jump enabling operator applied to a process term p with set J^+ ($\iota_{J^+}(p)$) behaves the same as its argument in an environment where the variables from set J^+ are jumping variables.

$$\frac{(C, J \cup J^+, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\alpha} \langle \begin{matrix}\checkmark\\ p'\end{matrix}, \sigma'\rangle}{(C, J, L, H, R) \Vdash \langle \iota_{J^+}(p), \sigma\rangle \xrightarrow{\alpha} \langle \begin{matrix}\checkmark\\ \iota_{J^+}(p')\end{matrix}, \sigma'\rangle}\ 41$$

$$\frac{(C, J \cup J^+, L, H, R) \Vdash \langle p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle p', \sigma'\rangle}{(C, J, L, H, R) \Vdash \langle \iota_{J^+}(p), \sigma\rangle \overset{t,\rho}{\longmapsto} \langle \iota_{J^+}(p'), \sigma'\rangle}\ 42$$

$$\frac{(C, J \cup J^+, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}}{(C, J, L, H, R) \Vdash \langle \iota_{J^+}(p), \sigma\rangle \overset{\xi}{\rightsquigarrow}}\ 43$$

3.4.11 Variable scope operator

By means of the variable scope operator, local variables are introduced in a χ process. A variable scope operator process term

$$[\![_{\mathrm{V}}\, \sigma_{\mathrm{dx}_\perp}, \{\mathbf{x}\}, \{\mathbf{g}\} \mid p \,]\!],$$

that is used in an environment (C, J, L, H, R), with valuation σ, and where $\sigma_{\mathrm{dx}_\perp}$ denotes a local valuation that may have undefined values and that has domain $\{\mathbf{d}, \mathbf{x}\}$, $\mathbf{d}$ denotes the local discrete variables $d_1, \ldots, d_k$, $\mathbf{x}$ denotes the local (non-jumping) continuous variables $x_1, \ldots, x_n$, and $\mathbf{g}$ denotes the local algebraic variables $g_1, \ldots, g_m$, behaves as p after taking the union of the respective categories (discrete, continuous and algebraic) of local and global variables and taking the union of the local and global valuation. To ensure that all local variables are fresh with respect to the global variables, the local variables are first renamed. Thus $\mathbf{d}', \mathbf{x}', \mathbf{g}'$, in the rules below, denote fresh variables $d'_1, \ldots, d'_k$, $x'_1, \ldots, x'_n$, $g'_1, \ldots, g'_m$ with respect to $\mathrm{dom}(\sigma) \cup L \cup \{\mathbf{d}\} \cup \{\mathbf{x}\} \cup \{\mathbf{g}\}$. Notation $p[\mathbf{d}', \mathbf{x}', \mathbf{g}'/\mathbf{d}, \mathbf{x}, \mathbf{g}]$ denotes the process term that is obtained by substitution of the (free) variables $\mathbf{d}, \mathbf{x}, \mathbf{g}$ in p by the fresh variables $\mathbf{d}', \mathbf{x}', \mathbf{g}'$, respectively, choosing the fresh variables $\mathbf{d}', \mathbf{x}', \mathbf{g}'$ in such a way that they remain free in p. After execution of an action or a delay transition, the local variables of the variable scope operator are renamed back to their original names. Note that the variables used in the recursion definitions R are not renamed to ensure that the bindings of these variables remain unchanged. In this way, the variables occurring in recursion definitions are bound statically, as is illustrated by the following example χ process:

$$\begin{array}{l}\langle [\![_{\mathrm{V}}\, \{n \mapsto 2\},\, \emptyset,\, \emptyset \mid X;\; z := n \,]\!] \\ ,\, \{n \mapsto 0,\; y \mapsto 0,\; z \mapsto 0\} \\ ,\, (\emptyset, \emptyset, \emptyset, \emptyset, \{X \mapsto n := 1;\; y := n\}) \\ \rangle.\end{array}$$

The process defines the variables n, y, z that are initialized to 0, a recursion definition $X \mapsto n := 1;\; y := n$, and a variable scope operator that redefines n as a local variable that is initialized to 2. When the process term $X;\; z := n$ terminates, the value of y equals 1, and the value of z equals 2. The recursion variable X is executed in the scope of its definition.

The variable scope operator is the only operator that affects the set of continuous variables C and the set of algebraic variables L from the environment. In this way, it is ensured that the discrete, continuous, or algebraic variables in any χ process $\langle p, \sigma, E\rangle$ remain discrete, continuous, or algebraic, respectively. Continuous variables, on the other hand, can change from non-jumping continuous variables to jumping continuous variables, using the jump enabling operator (see Section 3.4.10).

The local variables are invisible outside of the scope operator. This is done by means of data abstraction. For action transitions, data abstraction takes place by restricting the extended valuations, and the valuation of the resulting process, to the global variables, and by keeping only the global variables in the set W of the internal receive actions. For time transitions, data abstraction takes place by restricting the trajectory to the global variables. In this way, all changes to local variables are removed.

Action transition abstraction function $\kappa \in \Sigma \times \mathcal{P}(\dot{\mathcal{V}}) \times \mathcal{P}(\mathcal{V}) \times \dot{\Sigma} \times \mathcal{A} \times \dot{\Sigma} \to \dot{\Sigma} \times \mathcal{A} \times \dot{\Sigma}$ is defined as follows. For arbitrary receive actions $\mathrm{ira}(h, cs, W)$:

$$\kappa_{\sigma\dot{C}L}(\xi, \mathrm{ira}(h, cs, W), \xi') = \xi_{\sigma\dot{C}L},\ \mathrm{ira}(h, cs, W \cap (\mathrm{dom}(\sigma) \cup L)),\ \xi'_{\sigma\dot{C}L},$$

and for all other actions:

$$\kappa_{\sigma\dot{C}L}(\xi, a, \xi') = \xi_{\sigma\dot{C}L}, a, \xi'_{\sigma\dot{C}L},$$

where extended valuations $\xi_{\sigma\dot{C}L}$ and $\xi'_{\sigma\dot{C}L}$ denote $\xi \restriction (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)$ and $\xi' \restriction (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)$, respectively. Furthermore, in the rules below, the following abbreviations are used: valuation σ'_σ denotes $\sigma' \restriction \mathrm{dom}(\sigma)$, and trajectory $\rho_{\sigma\dot{C}L}$ denotes $\rho \downarrow (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)$.

Valuation $\sigma_{\mathrm{dx}_\perp} \in \{\mathbf{d}, \mathbf{x}\} \mapsto (\Lambda \cup \{\perp\})$ and valuation $\sigma_{\mathrm{d'x'}} \in \{\mathbf{d}', \mathbf{x}'\} \mapsto \Lambda$ define the same values for all (renamed) variables for which $\sigma_{\mathrm{dx}_\perp}$ is defined. For the undefined variables in $\sigma_{\mathrm{dx}_\perp}$, $\sigma_{\mathrm{d'x'}}$ has an arbitrary value: $\forall_{v \in \mathrm{dom}(\sigma_{\mathrm{dx}_\perp})}\, \sigma_{\mathrm{dx}_\perp}(v) \neq \perp \Rightarrow \sigma_{\mathrm{d'x'}}(v[\mathbf{d}', \mathbf{x}'/\mathbf{d}, \mathbf{x}]) = \sigma_{\mathrm{dx}_\perp}(v)$, where $v[\mathbf{d}', \mathbf{x}'/\mathbf{d}, \mathbf{x}]$ denotes the renamed version of variable v.

$$\frac{(C \cup \{\mathbf{x}'\}, J, L \cup \{\mathbf{g}'\}, H, R) \Vdash \langle p[\mathbf{d}', \mathbf{x}', \mathbf{g}'/\mathbf{d}, \mathbf{x}, \mathbf{g}], \sigma \cup \sigma_{\mathrm{d'x'}} \rangle \xrightarrow{\xi, a, \xi'} \langle \begin{matrix} \checkmark \\ p' \end{matrix}, \sigma' \rangle}{(C, J, L, H, R) \Vdash \langle [\![_{\mathrm{V}}\, \sigma_{\mathrm{dx}_\perp}, \{\mathbf{x}\}, \{\mathbf{g}\} \mid p\,]\!], \sigma \rangle \xrightarrow{\kappa_{\sigma\dot{C}L}(\xi, a, \xi')}} 44$$

$$\left\langle \begin{matrix} \checkmark \\ [\![_{\mathrm{V}}\, (\sigma' \restriction \{\mathbf{d}', \mathbf{x}'\})[\mathbf{d}, \mathbf{x}/\mathbf{d}', \mathbf{x}'], \{\mathbf{x}\}, \{\mathbf{g}\} \mid p'[\mathbf{d}, \mathbf{x}, \mathbf{g}/\mathbf{d}', \mathbf{x}', \mathbf{g}']\,]\!] \end{matrix}, \sigma'_\sigma \right\rangle$$

$$\frac{(C \cup \{\mathbf{x}'\}, J, L \cup \{\mathbf{g}'\}, H, R) \Vdash \langle p[\mathbf{d}', \mathbf{x}', \mathbf{g}'/\mathbf{d}, \mathbf{x}, \mathbf{g}], \sigma \cup \sigma_{\mathrm{d'x'}} \rangle \overset{t, \rho}{\longmapsto} \langle p', \sigma' \rangle}{(C, J, L, H, R) \Vdash \langle [\![_{\mathrm{V}}\, \sigma_{\mathrm{dx}_\perp}, \{\mathbf{x}\}, \{\mathbf{g}\} \mid p\,]\!], \sigma \rangle \overset{t, \rho_{\sigma\dot{C}L}}{\longmapsto}} 45$$

$$\langle [\![_{\mathrm{V}}\, (\sigma' \restriction \{\mathbf{d}', \mathbf{x}'\})[\mathbf{d}, \mathbf{x}/\mathbf{d}', \mathbf{x}'], \{\mathbf{x}\}, \{\mathbf{g}\} \mid p'[\mathbf{d}, \mathbf{x}, \mathbf{g}/\mathbf{d}', \mathbf{x}', \mathbf{g}']\,]\!], \sigma'_\sigma \rangle$$

$$\frac{(C \cup \{\mathbf{x}'\}, J, L \cup \{\mathbf{g}'\}, H, R) \Vdash \langle p[\mathbf{d}', \mathbf{x}', \mathbf{g}'/\mathbf{d}, \mathbf{x}, \mathbf{g}], \sigma \cup \sigma_{\mathrm{d'x'}} \rangle \overset{\xi}{\rightsquigarrow}}{(C, J, L, H, R) \Vdash \langle [\![_{\mathrm{V}}\, \sigma_{\mathrm{dx}_\perp}, \{\mathbf{x}\}, \{\mathbf{g}\} \mid p\,]\!], \sigma \rangle \overset{\xi_{\sigma\dot{C}L}}{\rightsquigarrow}} 46$$

3.4.12 Channel scope operator

By means of the channel scope operator, local channels can be introduced in a χ process. By means of action abstraction, communication actions on local channels are made invisible outside of the scope operator.

Action abstraction takes place by substituting communication actions $\mathrm{ca}(h, cs)$ using a local channel by internal τ actions (see Rule 47). The internal send and receive actions ($\mathrm{isa}(h, cs)$ and $\mathrm{ira}(h, cs, W)$) on a local channel h are blocked, because Rule 47 only specifies behavior for communication actions $\mathrm{ca}(h, cs)$. Therefore, these internal send and receive actions are not visible outside of the scope operator. Function $\mathrm{ch} \in \mathcal{A} \to \mathcal{H} \cup \{\bot\}$ extracts the channel label from an action. It is defined as $\mathrm{ch}(\mathrm{ca}(h, cs)) = h$, $\mathrm{ch}(\mathrm{isa}(h, cs)) = h$, $\mathrm{ch}(\mathrm{ira}(h, cs, W)) = h$, and $\mathrm{ch}(l_\mathrm{a}) = \bot$, where $l_\mathrm{a} \in A_\mathrm{label}$. Note that no renaming is applied to action a in Rule 48, because this action cannot refer to local channels.

The local channels $\mathbf{h}$ occurring in p are renamed to fresh channels $\mathbf{h}'$ in a similar way as for the local variables in the variable scope operator. Also here, in the channel scope operator, renaming does not take place in the recursion definitions R to ensure that the bindings of channels in R remain unchanged.

$$\frac{(C, J, L, H \cup \{\mathbf{h}'\}, R) \Vdash \langle p[\mathbf{h}'/\mathbf{h}], \sigma\rangle \xrightarrow{\xi, \mathrm{ca}(h,cs), \xi'} \langle \begin{matrix}\checkmark \\ p'\end{matrix}, \sigma'\rangle,\ h \in \{\mathbf{h}'\}}{(C, J, L, H, R) \Vdash \langle [\![_\mathrm{H}\, \{\mathbf{h}\} \mid p\,]\!], \sigma\rangle \xrightarrow{\xi, \tau, \xi'} \langle \begin{matrix}\checkmark \\ [\![_\mathrm{H}\, \{\mathbf{h}\} \mid p'[\mathbf{h}/\mathbf{h}']\,]\!]\end{matrix}, \sigma'\rangle} \quad 47$$

$$\frac{(C, J, L, H \cup \{\mathbf{h}'\}, R) \Vdash \langle p[\mathbf{h}'/\mathbf{h}], \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle \begin{matrix}\checkmark \\ p'\end{matrix}, \sigma'\rangle,\ \mathrm{ch}(a) \notin \{\mathbf{h}'\}}{(C, J, L, H, R) \Vdash \langle [\![_\mathrm{H}\, \{\mathbf{h}\} \mid p\,]\!], \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle \begin{matrix}\checkmark \\ [\![_\mathrm{H}\, \{\mathbf{h}\} \mid p'[\mathbf{h}/\mathbf{h}']\,]\!]\end{matrix}, \sigma'\rangle} \quad 48$$

$$\frac{(C, J, L, H \cup \{\mathbf{h}'\}, R) \Vdash \langle p[\mathbf{h}'/\mathbf{h}], \sigma\rangle \overset{t,\rho}{\longmapsto} \langle p', \sigma'\rangle}{(C, J, L, H, R) \Vdash \langle [\![_\mathrm{H}\, \{\mathbf{h}\} \mid p\,]\!], \sigma\rangle \overset{t,\rho}{\longmapsto} \langle [\![_\mathrm{H}\, \{\mathbf{h}\} \mid p'[\mathbf{h}/\mathbf{h}']\,]\!], \sigma'\rangle} \quad 49$$

$$\frac{(C, J, L, H \cup \{\mathbf{h}'\}, R) \Vdash \langle p[\mathbf{h}'/\mathbf{h}], \sigma\rangle \overset{\xi}{\rightsquigarrow}}{(C, J, L, H, R) \Vdash \langle [\![_\mathrm{H}\, \{\mathbf{h}\} \mid p\,]\!], \sigma\rangle \overset{\xi}{\rightsquigarrow}} \quad 50$$

3.4.13 Recursion scope operator

By means of the recursion scope operator, local recursion definitions are introduced in a χ process. The application of the recursion scope operator to a process term p with a 'global' valuation σ and a 'global' environment (C, J, L, H, R) behaves as p after the addition of local recursion definitions to the global recursion definitions. In the rules below, $\mathbf{X} \mapsto \mathbf{q}$ denotes the recursion definitions $X_1 \mapsto q_1, \ldots, X_r \mapsto q_r$. To prevent redefinition of recursion definitions already existing in the environment, the local recursion variables $\mathbf{X}$ are renamed to fresh variables $\mathbf{X}'$ with respect to the variables from the domain of R.

$$\frac{(C, J, L, H, R \cup \{\mathbf{X}' \mapsto \mathbf{q}[\mathbf{X}'/\mathbf{X}]\}) \Vdash \langle p[\mathbf{X}'/\mathbf{X}], \sigma\rangle \xrightarrow{\alpha} \langle \begin{matrix}\checkmark \\ p'\end{matrix}, \sigma'\rangle}{(C, J, L, H, R) \Vdash \langle [\![_{\mathrm{R}}\, \{\mathbf{X} \mapsto \mathbf{q}\} \mid p\,]\!], \sigma\rangle \xrightarrow{\alpha} \langle \begin{matrix}\checkmark \\ [\![_{\mathrm{R}}\, \{\mathbf{X} \mapsto \mathbf{q}\} \mid p'[\mathbf{X}/\mathbf{X}']\,]\!]\end{matrix}, \sigma'\rangle} \; 51$$

$$\frac{(C, J, L, H, R \cup \{\mathbf{X}' \mapsto \mathbf{q}[\mathbf{X}'/\mathbf{X}]\}) \Vdash \langle p[\mathbf{X}'/\mathbf{X}], \sigma\rangle \xmapsto{t,\rho} \langle p', \sigma'\rangle}{(C, J, L, H, R) \Vdash \langle [\![_{\mathrm{R}}\, \{\mathbf{X} \mapsto \mathbf{q}\} \mid p\,]\!], \sigma\rangle \xmapsto{t,\rho} \langle [\![_{\mathrm{R}}\, \{\mathbf{X} \mapsto \mathbf{q}\} \mid p'[\mathbf{X}/\mathbf{X}']\,]\!], \sigma'\rangle} \; 52$$

$$\frac{(C, J, L, H, R \cup \{\mathbf{X}' \mapsto \mathbf{q}[\mathbf{X}'/\mathbf{X}]\}) \Vdash \langle p[\mathbf{X}'/\mathbf{X}], \sigma\rangle \overset{\xi}{\rightsquigarrow}}{(C, J, L, H, R) \Vdash \langle [\![_{\mathrm{R}}\, \{\mathbf{X} \mapsto \mathbf{q}\} \mid p\,]\!], \sigma\rangle \overset{\xi}{\rightsquigarrow}} \; 53$$

Consider, for example, the process term $[\![_{\mathrm{R}}\, X \mapsto Y, Y \mapsto x := 0 \mid [\![_{\mathrm{R}}\, Y \mapsto x := 1 \mid X\,]\!]\!]$. Local recursion variable Y with definition $Y \mapsto x := 1$ conflicts with the recursion variable definition $Y \mapsto x := 0$ from the outer scope. The renaming of the local variable in the rules of the recursion scope operator ensures that the process term behaves as $[\![_{\mathrm{R}}\, X \mapsto Y, Y \mapsto x := 0 \mid [\![_{\mathrm{R}}\, Z \mapsto x := 1 \mid X\,]\!]\!]$. Thus, the value of variable x becomes 0.

4 Examples

This section presents three examples. Many additional examples can be found in [2].

4.1 Constrained pendulum

Figure 1 shows a constrained pendulum that is also defined in [9,26]. The equations of motion of this pendulum are given by Equation 1. The angle

between the pendulum and the vertical is denoted by θ, ω denotes the angular velocity of the pendulum, and l denotes the distance between the rotation point and the mass.

$$\begin{aligned} \dot{\theta} &= \omega \\ ml\dot{\omega} &= -mg\sin(\theta) - dl\omega \end{aligned} \tag{1}$$

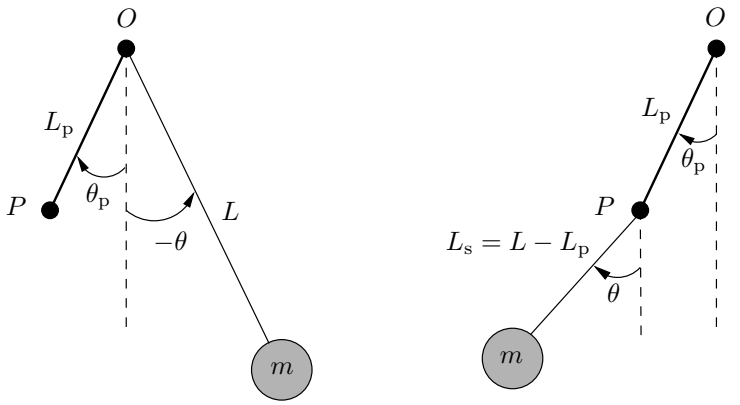

Fig. 1. Constrained Pendulum.

The mass and maximum length of the pendulum are represented by m and L, respectively. The damping coefficient and the acceleration due to gravity are denoted by d and g. The angle of the constraint is denoted by θ_p. In order to keep the example as small and clear as possible, it is assumed that $\theta_\mathrm{p} \geq 0$ and $|\theta| \leq \pi/2$. Also, it is assumed that the pendulum always remains in a straight line from the rotation point to the end point. The χ model is:

$\langle$ cont θ, ω, alg l
, $\theta = \theta_0, \omega = \omega_0$
, long $\mapsto l = L,\ \theta \leq \theta_\mathrm{p}$ [] $[\omega := \frac{L}{L_\mathrm{s}}\omega]$; short
, short $\mapsto l = L_\mathrm{s},\ \theta \geq \theta_\mathrm{p}$ [] $[\omega := \frac{L_\mathrm{s}}{L}\omega]$; long
| (skip; long [] skip; short)
|| $\dot{\theta} = \omega$
, $ml\dot{\omega} = -mg\sin(\theta) - dl\omega$
$\rangle$,

where θ_0 and ω_0 denote constants representing the initial values of θ and ω, respectively. When $\theta \leq \theta_\mathrm{p}$ or $\theta \geq \theta_\mathrm{p}$, the pendulum can delay in mode long or short, respectively. In mode long, the assignment $\omega := \frac{L}{L_\mathrm{s}}\omega$ can be executed only if the new state after the assignment to ω is consistent with the constraints $l = L_\mathrm{s},\ \theta \geq \theta_\mathrm{p}$ of mode short, because a process cannot enter an inconsistent state. Therefore, mode switches are possible only for $\theta = \theta_\mathrm{p}$. The any delay operator applied on the assignment in $[\omega := \frac{L}{L_\mathrm{s}}\omega]$ is needed, because otherwise the assignment and the alternative composition would not be able

to delay. Note that the model allows infinite switching between modes long and short, without progress of time, when $\theta = \theta_{\mathrm{p}}$. This switching behavior can, in principle, be avoided by guarding the delayable assignments $[\omega := \frac{L}{L_{\mathrm{s}}}\omega]$ and $[\omega := \frac{L_{\mathrm{s}}}{L}\omega]$ with (non-trivial) conditions that prevent mode switching when no delay behavior is possible in the new mode.

4.2 Glider take-off

Figure 2 shows a glider that is towed off the ground by a tow plane. The position, velocity and acceleration of the tow plane are given by x_1, v_1, a, respectively. The position and velocity of the glider are given by x_2 and v_2.

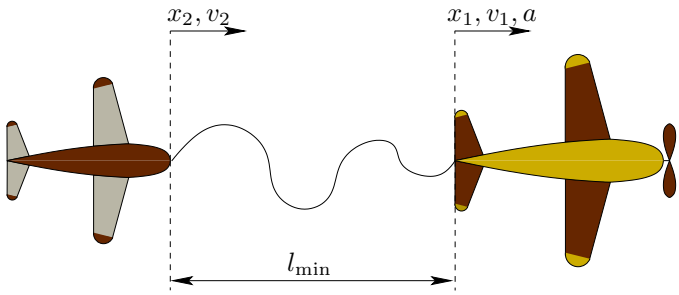

Fig. 2. Glider take-off.

Initially, the tow plane and glider are standing still at a distance of $l_{\min}$. After one unit of time, the tow plane very slowly accelerates ($a := 0.02$) until the tow cable is at its maximum length of $l_{\max}$. At that moment, the velocity of the glider jumps discontinuously to the velocity of the tow plane. We assume the mass of the glider to be considerably smaller than the mass of the tow plane. The tow plane then accelerates ($a := 0.5$) until its velocity is at $v_{\max}$. After another t units of time, the glider releases the tow cable, and continues on its own. Its velocity is then assumed to be determined by the air resistance, which is proportional to the squared velocity (kv_2^2), and the propelling forces F, which we assume constant in the model below:

$$
\begin{array}{l}
\langle\ \mathsf{disc}\ s, a, \mathsf{cont}\ x_1, x_2, v_1, v_2 \\
,\ s = \textsc{stop}, a = 0, x_1 = l_{\min}, x_2 = 0, v_1 = 0 \\
\mid\ \dot{x}_1 = v_1,\ \dot{x}_2 = v_2,\ \dot{v}_1 = m_1 a \\
\parallel\ s = \textsc{stop}\ \ \rightarrow v_2 = 0 \\
\parallel\ s = \textsc{tow}\ \ \rightarrow v_2 = v_1 \\
\parallel\ s = \textsc{fly}\ \ \rightarrow \dot{v}_2 = F - kv_2^2 \\
\parallel\ \Delta 1;\ a := 0.02 \\
;\ x_1 - x_2 \geq l_{\max} \rightarrow (\mathsf{jump}\ v_2 \mid s := \textsc{tow}) \\
;\ a := 0.5;\ v_1 \geq v_{\max} \rightarrow a := 0 \\
;\ \Delta t;\ s := \textsc{fly}
\end{array}
$$

⟩

In the model, m_1 is a constant denoting the mass of the towing plane, k is some constant, and enumeration variable s denotes the state of the glider. When the distance between the tow plane and the glider becomes equal to the maximum length of the cable, the glider abruptly starts moving. This is modeled by (jump v_2 | $s :=$ TOW). The jump enabling operator (jump v_2 | ...) enables jumps for continuous variable v_2 when assignment $s :=$ TOW) is executed. This is necessary, because v_2 is declared as a (non-jumping) continuous variable. The only assignment where v_2 must be able to jump is the assignment $s :=$ TOW, because then v_2 must discontinuously change to the value of v_1 in order to satisfy equation $v_2 = v_1$ that must hold for $s =$ TOW. In this example, the relation $v_2 = v_1$ in mode TOW is so straightforward, that the jumping behavior of variable v_2 when mode TOW becomes active can also be modeled explicitly by means of a multi-assignment $(s, v_2 :=$ TOW$, v_1)$ instead of (jump v_2 | $s :=$ TOW). The model with the jump enabling operator is more general, because it can also be used in cases where the algebraic constraints are so complex that it becomes difficult, or impossible, to explicitly calculate the new value of the jumping variable after the discontinuity.

4.3 Bottle filling system

The bottle filling system from Figure 3 consists of a liquid storage tank, and two identical bottle filling lines.

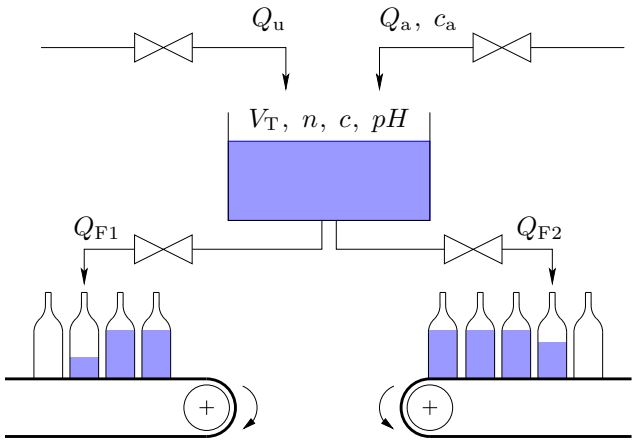

Fig. 3. The bottle filling system.

The bottles are filled with liquid from the storage tank. A control system keeps the volume V_{T} in the storage tank between 2 and 10, and the pH level (acidity) of the liquid in the storage tank between 7 and 7.1. The liquid in the storage tank slowly becomes less acidic (pH level increases). To correct this, a strong acid is dribbled into the storage tank when the acidity of the liquid becomes too low ($pH \geq 7.1$).

Figure 4 shows the iconic model of the bottle filling system. The lines ending in a small circle represent shared variables (V_T, Q_{F_1}, Q_{F_1}).

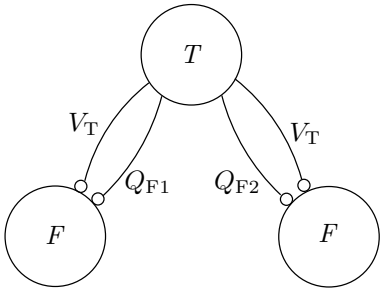

Fig. 4. Iconic representation of the bottle filling system model.

The acid and liquid supply processes are not modeled, since we consider the acid and liquid always to be available, and we are not interested in the amount of acid or liquid that is used. The χ specification of the bottle filling system is as follows:

$$
\begin{array}{l}
\langle\ \mathsf{cont}\ V_\mathrm{T},\ \mathsf{alg}\ Q_{\mathrm{F1}}, Q_{\mathrm{F2}} \\
,\ V_\mathrm{T} = 2 \\
|\ T(V_\mathrm{T}, Q_{\mathrm{F1}}, Q_{\mathrm{F2}}) \\
\|\ F(V_\mathrm{T}, Q_{\mathrm{F1}}) \\
\|\ F(V_\mathrm{T}, Q_{\mathrm{F2}}) \\
\rangle
\end{array}
$$

The storage tank and the two bottle filling lines are connected by means of the variables Q_{F1}, and Q_{F2}, respectively. Since a bottle may start filling only if the storage tank contains at least a volume of 0.7, the volume V_T of the storage tank is available in both bottle filling processes.

The molar quantity and molar concentration of the acid in the storage tank are denoted by n and c, respectively, where $n = cV$. The incoming flows of liquid and acid of the liquid storage tank T are denoted by Q_u and Q_a, respectively. The outgoing flows to the two bottle filling processes are denoted by Q_{F1} and Q_{F2}, respectively.

It is assumed that the liquids are incompressible, and that the volumes of the fluids remain the same when they are mixed. In such a case, the volume V of the mixed liquid equals the sum of its components which leads to the following equation

$$\dot{V} = Q_\mathrm{u} + Q_\mathrm{a} - Q_{\mathrm{F1}} - Q_{\mathrm{F2}}.$$

Next, the mass balance (actually mol balance) for the dissolved substance is derived. Acid comes into the tank by means of the flows Q_u and Q_a. Acid leaves the tank in outgoing flows Q_{F1} and Q_{F2}. Because the concentrations are in [mol/m^3], they can be directly multiplied with the flows (in [m^3/s]),

which leads to

$$\dot{n} = c_{\mathrm{u}} Q_{\mathrm{u}} + c_{\mathrm{a}} Q_{\mathrm{a}} - cQ_{\mathrm{F1}} - cQ_{\mathrm{F2}},$$

where c_{u} and c_{a} denote the concentrations of acid in the flows Q_{u} and Q_{a}. The gradual reduction of the acidity of the liquid is modeled by means of a constant K_{loss}, which leads to

$$\dot{n} = c_{\mathrm{u}} Q_{\mathrm{u}} + c_{\mathrm{a}} Q_{\mathrm{a}} - cQ_{\mathrm{F1}} - cQ_{\mathrm{F2}} - K_{\mathrm{loss}} V.$$

It is assumed that the acid is completely decomposed. Taking into account that the units of c are in [mol/m^3] instead of [mol/l], the pH is given by

$$pH = -\log c/1000.$$

The χ specification of the liquid storage tank follows below, where symbols Q_{seta}, Q_{setu}, c_{a}, c_{u}, and K_{loss} denote constants:

$$\begin{array}{l}
T(\mathsf{ext}\ V, Q_{\mathrm{F1}}, Q_{\mathrm{F2}}) \\
[\![\ \mathsf{disc}\ \alpha, \beta,\ \mathsf{cont}\ n,\ \mathsf{alg}\ pH, c, Q_{\mathrm{a}}, Q_{\mathrm{u}} \\
,\ \alpha = 0, \beta = 0, pH = 7 \\
|\quad \dot{V} = Q_{\mathrm{u}} + Q_{\mathrm{a}} - Q_{\mathrm{F1}} - Q_{\mathrm{F2}} \\
,\quad \dot{n} = c_{\mathrm{u}} Q_{\mathrm{u}} + c_{\mathrm{a}} Q_{\mathrm{a}} - cQ_{\mathrm{F1}} - cQ_{\mathrm{F2}} - K_{\mathrm{loss}} V \\
,\quad n = cV \\
,\ pH = -\log c/1000 \\
,\ Q_{\mathrm{a}} = \alpha Q_{\mathrm{seta}} \\
,\ Q_{\mathrm{u}} = \beta Q_{\mathrm{setu}} \\
\|\ *(\ pH \geq 7.1 \rightarrow \alpha := 1;\ pH \leq 7 \rightarrow \alpha := 0\ ) \\
\|\ *(\ V \leq 2 \rightarrow \beta := 1;\ V \geq 10 \rightarrow \beta := 0\ ) \\
]\!]
\end{array}$$

The model of the liquid storage tank T illustrates that a differential variable, such as variable n, is not necessarily initialized. In this case, instead, the algebraic variable pH is initialized ($pH = 7$). The continuous variables of the bottle filling system with tank T, can be declared in different ways.

In most cases, the differential variables, in this case V and n, are declared as (non-jumping) continuous variables. The other variables, not occurring with a dot (derivative) are then declared as algebraic variables. This ensures that the differential variables can be assigned new values, causing discontinuities. The algebraic variables will then simultaneously jump to their new values satisfying the equations. This declaration scheme is used in process T. Note that variable V is an external variable that is declared as a (non-jumping) continuous variable in the preceding χ process that defines the complete bottle

filling system. Note that even though pH is an algebraic variable, which is not normally assigned new values, pH can be initialized, in this case to a value of 7, in the initialization predicate.

In process T, the only discontinuities in continuous variables occur in the flows Q_{F1}, Q_{F2}, Q_{a}, and Q_{u}, that are switched on and off discontinuously in process T, and in process F that follows below. Therefore, the algebraic variables apart from these flows could just as well have been declared as (non-jumping) continuous variables as in $\mathsf{cont}\ n, pH, c$.

The behavior of the model is explained as follows. Initially, the pH of the liquid in the storage tank equals 7. It is assumed that the pH level of the incoming liquid is 7 or more, since the acidity controller can only make the acidity of the storage tank increase, causing the pH to decrease. If the pH value exceeds the maximum value ($pH \geq 7.1$), the acid valve is opened ($\alpha := 1$) so that acid is dribbled into the tank. Dribbling of the acid continues until the pH value comes back at 7, and the valve is closed ($\alpha := 0$). In a similar way, the controller tries to keep the level of the storage tank between 2 and 10.

The model of a bottle filling line follows below, where symbols Q_{setF}, and t_{tr} denote constants.

$$
\begin{array}{l}
F(\mathsf{ext}\ V_{\mathrm{T}}, Q_{\mathrm{F}}) = \\
[\![\ \mathsf{disc}\ \alpha,\ \mathsf{cont}\ V \\
,\ \alpha = 0, V = 0 \\
|\quad \dot{V} = Q_{\mathrm{F}} \\
,\ Q_{\mathrm{F}} = \alpha Q_{\mathrm{setF}} \\
\|\ *(\ V_{\mathrm{T}} \geq 0.7 \rightarrow \alpha := 1 \\
\quad ;\ \alpha = 1\ \stackrel{*}{\rightarrow}\ (\ V \geq 1 \rightarrow \alpha := 0 \\
\qquad\qquad\qquad [\!]\ V_{\mathrm{T}} \leq 0.5 \rightarrow \alpha := 0;\ V_{\mathrm{T}} \geq 0.7 \rightarrow \alpha := 1 \\
\qquad\qquad\qquad)\\
\quad ;\ \Delta t_{\mathrm{tr}};\ V := 0 \\
\quad)\\
]\!]
\end{array}
$$

The valve switching the flow Q_{F} is modeled by means of the discrete variable α. When the volume in the storage tank is at least 0.7, the bottle filling process can be started ($\alpha := 1$). Filling stops when the volume in the storage tank drops below 0.5 ($V_{\mathrm{T}} \leq 0.5 \rightarrow \alpha := 0$). Filling resumes when the volume in the storage tank is at least 0.7. Filling also stops when the bottle is full ($V \geq 1 \rightarrow \alpha := 0$). The time needed to place a new bottle under the filling

nozzle is given by t_{tr}. After that, the bottle volume is reset to 0, which models the arrival of a new bottle, and the filling process is repeated.

5 Validation of the semantics

First we consider the well-definedness of the semantics in Section 5.1. Then, in Section 5.2, some properties of the χ semantics are given. In Section 5.3, a notion of equivalence is defined, called stateless bisimilarity [27], which is similar to the well-known notion of bisimilarity [28,29]. It is also shown that this relation is an equivalence and a congruence for all χ operators. Some useful properties of closed χ process terms are given in Section 5.4. Many of these properties express intuitions about the meaning of the χ operators such as the commutativity and associativity of the alternative composition and the parallel composition operator. Other properties are introduced for the purpose of simplifying χ models. Both the examples treated in the previous section and the properties treated in this section add to the level of confidence one has with respect to the 'correctness' of the semantics.

5.1 Well-definedness of the semantics

In the term deduction system, negative premises are used in Rule 37 of the urgent communication operator. As a consequence it is not obvious at first sight whether the term deduction system defines a unique transition system for each closed process term. Well-definedness of the term deduction system can be obtained by providing a *stratification* [30]. The mapping that associates with every positive action transition and positive consistency predicate the value 0 and with every positive time transition the value 1, turns out to be a stratification.

5.2 Properties of the semantics

In this section, some useful properties about the semantics of χ are introduced. The proofs of these properties are given in Appendix A. The properties are applied in the remainder of the paper, especially in the proofs of the properties defined in Section 5.4.

With the current set of deduction rules for the semantics of χ, the left-hand (ξ) and right-hand (ξ') extended valuation restricted to the domain of σ are always the same as the initial (σ) and resulting (σ') valuation of an action

transition, respectively. A similar reasoning applies to the first and last valuation of a trajectory on a time transition and the initial and resulting valuation, respectively. Also note that the environment is never changed in a transition, and that the extended valuation in the consistency predicate restricted to the model variables is the same as the initial valuation.

The following lemma captures these facts.

Lemma 1 *Let p and p' be closed process terms, σ, σ' be valuations, ξ, ξ' be extended valuations, E and E' be environments, a be an action, ρ be a trajectory, and $t \in T$. Then*

$$\begin{aligned}
\langle p, \sigma, E\rangle \xrightarrow{\xi, a, \xi'} \langle _, \sigma', E'\rangle \Rightarrow\ & \mathrm{dom}(\sigma) = \mathrm{dom}(\sigma') \wedge \xi_\sigma = \sigma \wedge \xi'_{\sigma'} = \sigma' \\
& \wedge E = E', \\
\langle p, \sigma, E\rangle \xmapsto{t, \rho} \langle p', \sigma', E'\rangle \Rightarrow\ & \mathrm{dom}(\rho) = [0, t] \wedge \rho_\sigma(0) = \sigma \wedge \rho_{\sigma'}(t) = \sigma' \\
& \wedge E = E', \\
\langle p, \sigma, E\rangle \overset{\xi}{\rightsquigarrow} \Rightarrow\ & \xi_\sigma = \sigma.
\end{aligned}$$

where $\langle p, \sigma, E\rangle \xrightarrow{\xi, a, \xi'} \langle \_, \sigma', E'\rangle$ is an abbreviation for $\langle p, \sigma, E\rangle \xrightarrow{\xi, a, \xi'} \langle \overset{\checkmark}{p'}, \sigma', E'\rangle$ for some p'.

The χ processes that can perform action or time transitions are consistent (the consistency predicate holds).

Lemma 2 *Let p and p' be closed process terms, σ and σ' be valuations, E and E' be environments, ξ and ξ' be extended valuations and a be an action. Then*

$$\langle p, \sigma, E\rangle \xrightarrow{\xi, a, \xi'} \Rightarrow \langle p, \sigma, E\rangle \overset{\xi}{\rightsquigarrow},$$

where $\langle p, \sigma, E\rangle \xrightarrow{\xi, a, \xi'}$ is an abbreviation for $\exists_{p', \sigma', E'} \langle p, \sigma, E\rangle \xrightarrow{\xi, a, \xi'} \langle \overset{\checkmark}{p'}, \sigma', E'\rangle$.

Lemma 3 *Let p and p' be closed process terms, σ and σ' be valuations, E and E' be environments, $t \in T$, and ρ be a trajectory. Then,*

$$\langle p, \sigma, E\rangle \xmapsto{t, \rho} \Rightarrow \langle p, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow},$$

where $\langle p, \sigma, E\rangle \xmapsto{t, \rho}$ is an abbreviation for $\exists_{p', \sigma', E'} \langle p, \sigma, E\rangle \xmapsto{t, \rho} \langle p', \sigma', E'\rangle$.

The following lemma shows that any variation in the set of jumping variables

in the environment of a consistent χ process has no effect on the consistency predicate.

Lemma 4 *Let p be a closed process term, σ be a valuation, C, J, W, L be sets of various classes of χ variables such that J and $W \subseteq \mathrm{dom}(\sigma) \setminus \{\mathsf{time}\}$, H be a set of channels, R be a recursion definition, and ξ be an extended valuation. Then*

$$\langle p, \sigma, (C, J, L, H, R)\rangle \overset{\xi}{\rightsquigarrow} \Leftrightarrow \langle p, \sigma, (C, J \cup W, L, H, R)\rangle \overset{\xi}{\rightsquigarrow} .$$

5.3 Stateless bisimilarity

Two closed χ process terms are considered equivalent if they have the same behavior (in the bisimulation sense) in case both are considered from the same initial valuation of model variables and the same environment. We also assume that the initial valuation contains at least the free occurrences of variables in the two closed χ process terms being equivalent.

Definition 5 (Stateless bisimilarity) *A stateless bisimulation relation on closed process terms is a relation $R \subseteq P \times P$ such that for all $(p, q) \in R$, the following holds:*

(1) $\forall_{\sigma,E,\xi,a,\xi',\sigma',E'} \langle p, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma', E'\rangle$
$\Leftrightarrow \langle q, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma', E'\rangle,$

(2) $\forall_{\sigma,E,\xi,a,\xi',p',\sigma',E'} \langle p, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle p', \sigma', E'\rangle$
$\Rightarrow \exists_{q'} \langle q, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle q', \sigma', E'\rangle \wedge (p', q') \in R,$

(3) $\forall_{\sigma,E,\xi,a,\xi',q',\sigma',E'} \langle q, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle q', \sigma', E'\rangle$
$\Rightarrow \exists_{p'} \langle p, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle p', \sigma', E'\rangle \wedge (p', q') \in R,$

(4) $\forall_{\sigma,E,t,\rho,p',\sigma',E'} \langle p, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle p', \sigma', E'\rangle$
$\Rightarrow \exists_{q'} \langle q, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q', \sigma', E'\rangle \wedge (p', q') \in R,$

(5) $\forall_{\sigma,E,t,\rho,q',\sigma',E'} \langle q, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q', \sigma', E'\rangle$
$\Rightarrow \exists_{p'} \langle p, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle p', \sigma', E'\rangle \wedge (p', q') \in R,$

(6) $\forall_{\sigma,E,\xi} \langle p, \sigma, E\rangle \overset{\xi}{\rightsquigarrow} \Leftrightarrow \langle q, \sigma, E\rangle \overset{\xi}{\rightsquigarrow} .$

Two closed process terms p and q are stateless bisimilar, denoted by $p \underline{\leftrightarrow} q$, if there exists a stateless bisimulation relation R such that $(p, q) \in R$.

As a consequence of Lemma 1, the definition of stateless bisimilarity can be simplified considerably. Yet, with in mind future extensions of the χ formalism, it might well be the case that these properties of the semantics are lost. Since

we would prefer not to redo all the coming proofs (in such a future), this presentation was chosen.

Stateless bisimilarity is proved to be a congruence with respect to all χ operators. As a consequence, algebraic reasoning is facilitated, since it is allowed to replace equals by equals in any context.

Theorem 6 (Congruence) *Stateless bisimilarity is a congruence with respect to all χ operators.*

PROOF. The deduction rules of the χ formalism, satisfy the *process-tyft* format of [27]. Therefore, stateless bisimilarity is a congruence.

5.4 Properties of the Chi operators

In this section, some properties of the operators of χ that hold with respect to stateless bisimilarity are discussed. Most of these correspond well with our intuitions, and hence this can be considered as an additional validation of the semantics. It is not our intention to provide a complete list of such properties (complete in the sense that every equivalence between closed process terms is derivable from those properties). The proofs of the properties from this section are given in Appendix B.

Proposition 7 (Signal emission operator) *The following properties hold for all closed process terms $p \in P$ and predicates u, u':*

$\text{true} \curvearrowright p \ \underline{\leftrightarrow}\ p$	$u \curvearrowright u \ \underline{\leftrightarrow}\ u$
$\text{false} \curvearrowright p \ \underline{\leftrightarrow}\ \bot$	$u \curvearrowright (u' \curvearrowright p) \ \underline{\leftrightarrow}\ (u \wedge u') \curvearrowright p$

If a true predicate is emitted, the process term is simply executed. If falsity holds initially, the process term is inconsistent. There is no effect if a predicate u is emitted to itself. A concatenation of signal emissions leads to a signal emission with conjunction of predicates.

Proposition 8 (Alternative composition) *The following properties hold for all closed process terms $p, q, r \in P$:*

$p \parallel p \ \underline{\leftrightarrow}\ p$	$(p \parallel q) \parallel r \ \underline{\leftrightarrow}\ p \parallel (q \parallel r)$
$p \parallel q \ \underline{\leftrightarrow}\ q \parallel p$	

The alternative composition is idempotent, commutative and associative. The property $p \mathbin{[\!]} \delta \leftrightarrow p$ does not hold. Consider, for example $p = \text{true}$. Then $p \mathbin{[\!]} \delta$ cannot perform any time transitions, while p can perform arbitrary time transitions. Property $p \mathbin{[\!]} \delta \leftrightarrow \delta$ does not hold either. Consider, for example $p = \text{skip}$. Then $p \mathbin{[\!]} \delta$ can perform a τ transition, while δ cannot.

Proposition 9 (Guard operator) *The following properties hold for closed process terms $p \in P$ and guards b:*

$\text{true} \to p$	$\leftrightarrow\ p$	$b \to \bot$	$\leftrightarrow\ \neg b$
$\text{false} \to p$	$\leftrightarrow\ \text{true}$	$b \to (p \mathbin{[\!]} q)$	$\leftrightarrow\ b \to p \mathbin{[\!]} b \to q$

If a process term is guarded by a true predicate, the process term is simply executed. In case a process term is guarded by a false predicate, process term $\text{false} \to p$ can perform any time transition, hence equals a true predicate. An inconsistent process term that is guarded by any guard is equivalent to the negation of the guard. By rewriting this property as $u \leftrightarrow \neg u \to \bot$, where the delay predicate u and guard b share the same syntax, it becomes clear that the delay predicate is not a primitive. Finally, the guard distributes over the alternative composition operator.

Proposition 10 (Sequential composition) *The following properties hold for all closed process terms $p, q, r \in P$ and guards b:*

$\delta;\ p$	$\leftrightarrow\ \delta$	$(p \mathbin{[\!]} q);\ r$	$\leftrightarrow\ p;\ r \mathbin{[\!]} q;\ r$
$(p;\ q);\ r$	$\leftrightarrow\ p;\ (q;\ r)$	$b \to (p;\ q)$	$\leftrightarrow\ (b \to p);\ q$

A deadlock process term followed by some other process terms is equivalent to the deadlock process term itself since the deadlock process term does not terminate successfully, i.e., deadlock is a left-zero element for sequential composition. Sequential composition is associative and alternative composition distributes over sequential composition from the left. A guard distributes to the left argument of a sequential composition.

Proposition 11 (Parallel composition) *The following properties hold for all closed process terms $p, q, r \in P$:*

$p \parallel q$	$\leftrightarrow\ q \parallel p$	$(p \parallel q) \parallel r$	$\leftrightarrow\ p \parallel (q \parallel r)$

Parallel composition is commutative and associative.

Proposition 12 (Action encapsulation operator) *The following properties hold for all closed process terms $p \in P$, and sets of actions A, A':*

$\partial_\emptyset(p) \leftrightarrows p$	$\partial_A(\partial_{A'}(p)) \leftrightarrows \partial_{A \cup A'}(p)$

If there are no actions to be encapsulated, the application of the action encapsulation operator to a process term p has no effect. Multiple applications of the action encapsulation operator are equivalent to a single application where all the actions to be encapsulated are combined using union of sets of actions.

Proposition 13 (Inconsistent process) *The following properties hold for all closed process terms $p \in P$ and predicates u:*

$u \curvearrowright \bot$	$\leftrightarrows \bot$	$\bot ; p$	$\leftrightarrows \bot$
$p \mathbin{[\!]} \bot$	$\leftrightarrows \bot$	skip; $\bot$	$\leftrightarrows \delta$
$p \parallel \bot$	$\leftrightarrows \bot$	$\bot$	$\leftrightarrows$ false
$\partial_A(\bot)$	$\leftrightarrows \bot$		

The inconsistent process term is a zero element for the signal emission operator, alternative composition, parallel composition and the action encapsulation operator. It is also a left-zero element for sequential composition. Going on as $\bot$ after performing an action transition, for example skip, is impossible. Since $\bot$ and false predicate cannot perform any transition, both process terms are equivalent.

6 Related work

The χ formalism is a hybrid process algebra, and is thus related to the other hybrid process algebras: HyPA [31], process algebra for hybrid systems $\mathrm{ACP}_{\mathrm{hs}}^{\mathrm{srt}}$ [25], the ϕ-Calculus [32], the hybrid formalisms based on CSP [33,34], and the process algebra [35].

The latter three process algebras [33–35] differ from χ in that they do not have shared variables. Shared variables are essential for modular specification of continuous and hybrid systems. The two CSP based formalisms also differ from the other process algebras in that they use a denotational semantics instead of an operational semantics. An operational semantics is generally believed to be more intuitive and easier to understand than a denotational semantics [36].

The ϕ-Calculus differs from the other process algebras in that continuous behavior is not defined by means of predicates in process terms. Instead, continuous behavior is defined by means of an environment. Process terms operate on the environment to update initial values, vector-fields, and algebraic constraints. In this way, the ϕ-Calculus can deal with dynamically reconfigurable processes. The resulting differential equations are required to be autonomous. This limits the specification of continuous systems, using the ϕ-Calculus, to that of ordinary differential equations (ODEs).

The relation between χ, hybrid automata, HyPA and $\mathrm{ACP}_{\mathrm{hs}}^{\mathrm{srt}}$ is discussed below. When comparing χ to hybrid automata, it should be kept in mind that many different hybrid automaton definitions exist. Some definitions require solutions for the continuous variables to be differentiable functions, e.g. in [8,7]. Other definitions allow the more general case of piecewise differentiable or piecewise continuous functions, e.g. in [9]. In [10], for each variable a dynamic type can be defined, which allows among others solutions in the form of discontinuous functions. Most definitions of hybrid automata do not define urgent transitions, or they define urgent transitions in a restrictive way (non-guarded), as in [14]. In [5], urgent transitions are defined in a general way, using a predicate that defines the maximum sojourn time in a location. However, instead of invariants and flow clauses, evolution functions are used in locations. With respect to the meaning of jump clauses, that define the behavior of the variables in action transitions, differences also occur: where in [8] the variables can in principle perform arbitrary jumps unless restricted by the jump predicate, in [14], variables in principle remain unchanged unless changes are enforced by the jump predicate. Most hybrid automata distinguish between flow clauses, or vector fields, and invariants. In [37], however, invariants and flow clauses are combined into one predicate (as in $\mathrm{ACP}_{\mathrm{hs}}^{\mathrm{srt}}$, HyPA, and χ). Finally, some hybrid automata have a precisely defined syntax, in particular the input languages of the verification tools PHAver [38] and HyTech [14]. Many other hybrid automata are mainly semantical models, such as the hybrid automata defined in [10] and [11].

Where HyPA is a conservative extension of ACP from [39], and $\mathrm{ACP}_{\mathrm{hs}}^{\mathrm{srt}}$ is a conservative extension of a combination of the process algebra with continuous relative timing from [40] and the process algebra with propositional signals from [24], hybrid χ is not an extension of any previously existing process algebra. Hybrid χ has been proven to be an operational conservative extension of timed χ in [41]. The semantics of hybrid χ and timed χ, which is derived from hybrid χ, differs considerably from the semantics of their discrete-event predecessor χ_σ as defined in [19]. Where χ_σ has non-delayable guards, a weak time-deterministic alternative composition operator, urgent actions only, and no (global) time variable, the semantics of χ as defined in this paper has delayable guards, a strong time-deterministic alternative composition operator, urgent and non-urgent actions, and a global variable denoting the model time.

The integration between the DC and CS world views in χ was inspired by HyPA. Also, the use of delay predicates as atomic process term was inspired by HyPA. The χ formalism and $\text{ACP}_{\text{hs}}^{\text{srt}}$ were both strongly influenced by hybrid automata. $\text{ACP}_{\text{hs}}^{\text{srt}}$, χ, and hybrid automata share the 'consistent equation semantics'. For a hybrid automaton, the invariant of the current location should hold in the current state, and transitions to a new state and new location are allowed only if the invariant of the new location holds in the new state. Correspondingly, in $\text{ACP}_{\text{hs}}^{\text{srt}}$ and χ, the equations (delay predicates) of the process term should be consistent with the current state, and transitions to a new process term are allowed only if the equations (delay predicates) of the new process term are consistent with the new state. The hybrid automaton defined in [6] has a different semantics in that it allows transitions to a new location only if the invariant of the *current* location holds in the current state and in the new state. The signal emission operator in χ was inspired by the signal emission operator from $\text{ACP}_{\text{hs}}^{\text{srt}}$, which in its turn comes from the process algebra with relative timing from [40].

Some differences between χ, hybrid automata, $\text{ACP}_{\text{hs}}^{\text{srt}}$, and HyPA are:

- Where some hybrid automata and $\text{ACP}_{\text{hs}}^{\text{srt}}$ use continuous variables that are allowed to jump arbitrarily in an action transition with a true reset predicate, and other hybrid automata and HyPA use continuous variables that are not allowed to jump in an action transition, unless explicitly specified, χ uses both classes of continuous variables. Furthermore, χ adds discrete and algebraic variables. Some hybrid automata (e.g. see [11]) also define discrete variables (instead of locations). The behavior of the algebraic variables from χ is related to the external variables from the semantical hybrid automata defined in [10]. The external variables are not part of the state, and they can have a dynamic type that allows discontinuous trajectories. However, discrete transitions (action transitions) are defined only on internal variables, and the concept of internal and external variables is linked to visibility and hiding in [10]. In χ, all variables can be used in action predicates, and the different classes of variables and hiding/abstraction are orthogonal concepts.
- Where in $\text{ACP}_{\text{hs}}^{\text{srt}}$ and the hybrid automaton definition from [37] the dotted variables (derivatives) are part of the state (valuation), in HyPA, other hybrid automata, and χ they are not. The reason for this in χ is that the valuation together with the process term and the environment represent all that is needed to be able to determine future behavior. The values of the dotted variables are not needed for this purpose. For the same reason, algebraic variables are not part of the valuation in χ. Their values are determined completely by the process term.
- Where HyPA does not specify a solution concept for algebraic differential equations, and $\text{ACP}_{\text{hs}}^{\text{srt}}$ requires differentiability of the trajectories of the continuous variables, the χ semantics defines a solution concept that is pa-

rameterized with the type of trajectories allowed. In this paper, piecewise continuous functions for the trajectories of the algebraic and dotted variables are allowed. The parametrization of the solution concept in χ is related to the dynamic type present in [10]. Of course, since the solution concept of HyPA is a parameter of the semantics, it could use the solution concept defined in χ.

- Where in χ the passage of time cannot make a choice between the operands of alternative composition, in $\mathrm{ACP}_{\mathrm{hs}}^{\mathrm{srt}}$, the passage of time can enforce such a choice. In HyPA, the passage of time will always make a choice between the operands of the choice operator. This corresponds to the initial behavior of a hybrid automaton: depending on the initial state, a non-deterministic choice can be made for the first location where continuous behavior or discrete behavior may take place. After this first choice, a hybrid automaton cannot change location as a result of time passing, nor can outgoing edges disappear as a result of time passing.
- The syntactic extensions present in χ are unavailable in the other three formalisms, apart from the delay operator, which is also available in $\mathrm{ACP}_{\mathrm{hs}}^{\mathrm{srt}}$. However, in $\mathrm{ACP}_{\mathrm{hs}}^{\mathrm{srt}}$, the expression defining the amount of delay cannot contain variables. Furthermore, the scope operators, and process definition and instantiation process terms for complex system specification are available only in χ, apart from the variable scope operator which is added to HyPA in [42].

An interesting question is whether the χ functionality could have been obtained by extending HyPA and $\mathrm{ACP}_{\mathrm{hs}}^{\mathrm{srt}}$ with the χ scope operators, with the χ urgent communication operator, and with similar syntactic extensions as defined in χ. This approach suffers from fundamental limitations. The most important of these are:

- The algebraic variables present in χ cannot be incorporated in this way, because their functionality is reflected in the operational semantics of several process terms.
- The χ solution concept is quite different from the solution concept in $\mathrm{ACP}_{\mathrm{hs}}^{\mathrm{srt}}$.
- The semantics of the guards in χ (delayable) is fundamentally different from the semantics of the guards in HyPA and $\mathrm{ACP}_{\mathrm{hs}}^{\mathrm{srt}}$ (non-delayable).
- The flexibility of urgency in χ, where non-delayable actions have priority over delayable actions, is obtained by a carefully defined semantics of several operators (alternative composition, parallel composition, guard). It cannot be obtained by means of extensions to $\mathrm{ACP}_{\mathrm{hs}}^{\mathrm{srt}}$ or HyPA.
- The consistent equation semantics of χ is fundamentally different from the HyPA semantics, where equations can (temporarily) become inconsistent as a result of actions.

The additional functionality of χ makes axiomatization more difficult, when compared to $\mathrm{ACP}_{\mathrm{hs}}^{\mathrm{srt}}$ and HyPA. When it comes to tool support, the additional

functionality offered by χ probably means additional efforts for implementation. At this moment, it is difficult to further compare the expected efforts required for tool implementations of χ, $\text{ACP}_{\text{hs}}^{\text{srt}}$ and HyPA.

Other formalisms for hybrid system specification are hybrid Petri nets [43,44], and formalisms based on hybrid automata such as Charon [45] and Masaccio [46]. There are many differences and similarities between χ and these other formalisms. The main difference, however, between χ and other formalisms, including the process algebras and hybrid automata discussed before, is that we consider χ to be overall better suited to modeling. This may mean that certain phenomena can be modeled in χ whereas they cannot be modeled in another formalism, or that certain phenomena can be modeled more concisely or more intuitively in χ.

Which systems can be modeled in χ and not in other formalisms, or the other way round, is difficult to establish. It also depends on the notion of equivalence. For example, the equation $y = \text{step}(t - 1)$, where y is an algebraic variable, t denotes time and step is a discontinuous function such that $\text{step}(x)$ is 0 for $x < 0$ and 1 for $x \geq 0$, cannot be specified, or does not have the required behavior, in many formalisms. The required behavior can however be approximated by introducing an additional action to model the discontinuity. As another example, steady state initialization, as in $\dot{x} = 0 \curvearrowright \dot{x} = -x + 1$, cannot be expressed in most formalisms. When the equations are straightforward enough, however, the same effect can be obtained by direct initializations. In this example, by initializing variable x to 1.

The following properties make χ highly suited to modeling:

(1) The integration between the DC and CS world views as explained in Section 1. In this respect χ differs from the other formalisms mentioned above, apart from HyPA and the hybrid automata such as defined in [9].
(2) The combination of concise and intuitive language primitives, well suited to modeling, with a straightforward semantics, well suited to verification. This was in fact the biggest challenge in the design of χ. After numerous attempts to define the language primitives with associated syntax and semantics, it appeared that either the language was well suited to modeling, but with complex semantics, unsuited to verification; or the semantics was straightforward and elegant, but at the same time the language was cumbersome for modeling. The reason for this apparent contradiction is that the requirements for language primitives for verification and the requirements for language primitives for modeling are not the same.
(3) The relatively large number of operators dedicated to the modeling of discrete-event behavior: This makes it easy to abstract from continuous behavior and specify timed discrete-event models, without any continuous

variables and without differential (algebraic) equations. In this respect, χ has much in common with the hybrid formalisms based on CSP [33,34], and with $\text{ACP}_{\text{hs}}^{\text{srt}}$.

(4) Process instantiation, based on the modeling scope operator: this enables hierarchical composition of processes. It also provides encapsulation and data hiding, and it enables re-use of processes: parameterized processes can be defined once and instantiated many times with the same or different parameters. In this respect, the χ formalism is related to Charon and Masaccio, which allow components to be defined and instantiated. The χ formalism, being a process algebra, does not only allow parallel composition (as Charon and Masaccio) and sequential composition (as Masaccio), but allows in principle any combination of process terms by means of the χ operators.

Local variables, variable and/or action abstraction are present also in other formalisms. Hybrid I/O automata [10] define both action abstraction and variable abstraction, which are referred to as hiding of external actions and external variables. Hybrid (I/O) automata, however, need to be 'compatible' to allow parallel composition. Hybrid I/O automata, for example, require disjointness of the internal variables of the automata in parallel composition.

In χ, the concepts of variable abstraction and channel abstraction (comparable with action abstraction in other formalisms) are integrated in the modeling scope operator, which also provides a local scope for variables, channels, and recursion definitions. In this respect, the χ modeling scope operator is a high level modeling primitive unavailable in the other hybrid formalisms. Also, there are no compatibility restrictions on processes for parallel composition. Modular composition of processes is further supported by means of different interaction mechanisms. Processes can interact in three different ways:

- By means of shared variables, which is the main interaction mechanism for continuous-time processes consisting of systems of differential algebraic equations. Interaction between processes in Charon and Masaccio also takes place by means of shared variables. Synchronization by means of actions is, however, not supported.
- By means of channel based 'handshake synchronization'. It is comparable to actions in (hybrid) (I/O) automata and actions in ACP-based process algebras. A difference is that actions can be used to express synchronization between two or more processes. The synchronization mechanism used in χ is CSP [47] based. A channel can be shared by any number of processes, but synchronization always occurs on a point-to-point basis, so between exactly two processes. Another difference is that the interaction mechanism in χ also allows synchronous *communication*, as explained below, whereas actions are strictly used for synchronization.
- By means of synchronous communication, which is the CSP interaction

mechanism that combines synchronization with data-transfer, as also used in [33,34].

7 Conclusions and future work

The χ formalism differs considerably from other formalisms. On the one hand, it supports the dynamics and control way of hybrid systems modeling by means of discontinuous functions and/or switched equation systems, possibly leading to discontinuous trajectories. On the other hand, it supports the computer science way of hybrid systems modeling, where actions are used to model discontinuities. With respect to the computer science way of modeling, the χ formalism is heavily influenced by hybrid automata. The two formalisms both have a choice mechanism where, apart from initialization in a hybrid automaton, the passage of time cannot result in choices between operands (χ) or choices between locations or outgoing edges (hybrid automata). The χ formalism also shares the consistency concept with many hybrid automata: state changes in χ need to be consistent with delay predicates, which include the invariant and flow clauses of hybrid automata.

The χ formalism combines ease of modeling with a straightforward, formal semantics. Ease of modeling is ensured by means of different classes of variables, such as discrete, non-jumping continuous, jumping continuous and algebraic variables; by means of its delayable guard that ensures that the guard always holds when the first action of the guarded process term occurs; by means of its integration of urgent (non-delayable) and non-urgent (delayable) actions on the one hand, and urgent and non-urgent channels on the other hand; by means of allowing the modeling of differential algebraic equations as a process term as in mathematics; by means of allowing straightforward steady-state initialization; and by means of several user-friendly syntactic extensions.

The χ formalism is suited to modeling, simulation and verification of: (timed) discrete-event systems without (differential) equations, continuous-time systems consisting of ordinary differential equations with algebraic constraints, and combined discrete-event/continuous-time systems. It is especially suited to the specification and analysis of complex systems. This is achieved by means of the process terms for scoping, that integrate abstraction, local variables, local channels and local recursion definitions; by means of the process definition and instantiation syntactic extensions that enable process re-use, encapsulation, hierarchical and/or modular composition of processes; and by means of the different interaction mechanisms, namely handshake synchronization and synchronous communication that are mainly intended for discrete-event processes that do not share variables, and shared variables that are mainly intended for interaction between continuous-time or hybrid processes.

Future work entails among others:

- The definition of (formal) translations from χ to hybrid automata and input languages of verification tools to enable verification of model properties. For hybrid models PHAver [38], HyTech [7], CheckMate [48], `d/dt` [49], and the tools [50–53] are options.
- Redesign of the old hybrid χ simulator described in [17].
- Where $\mathrm{ACP}_{\mathrm{hs}}^{\mathrm{srt}}$ and HyPA have derived large sets of axioms to support equational reasoning, in χ, so far, only a limited set of properties has been derived. More properties need to be derived.

Acknowledgments

The authors would like to thank Jos Baeten, Pieter Cuijpers, Albert Hofkamp, Niek Jansen, Erjen Lefeber, Bas Luttik, Henk Nijmeijer, Sasha Pogromsky, and Frits Vaandrager for helpful comments and stimulating discussions. The authors also wish to thank the anonymous reviewers for their valuable comments.

References

[1] W. P. M. H. Heemels, B. D. Schutter, A. Bemporad, Equivalence of hybrid dynamical models, Automatica 37 (7) (2001) 1085–1091.

[2] D. A. van Beek, K. L. Man, M. A. Reniers, J. E. Rooda, R. R. H. Schiffelers, Syntax and consistent equation semantics of hybrid Chi, Tech. Rep. CS-Report 04-37, Eindhoven University of Technology, Department of Computer Science, The Netherlands (2004).

[3] A. F. Filippov, Differential Equations with Discontinuous Right Hand Sides, Kluwer Academic Publishers, Dordrecht, 1988.

[4] V. I. Utkin, Sliding Modes in Control Optimization, Springer-Verlag, Berlin, 1992.

[5] X. Nicollin, A. Olivero, J. Sifakis, S. Yovine, An approach to the description and analysis of hybrid systems, in: Workshop on Theory of Hybrid Systems, 1992, pp. 149–178.

[6] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P. H. Ho, X. Nicollin, A. Olivero, J. Sifakis, S. Yovine, The algorithmic analysis of hybrid systems, Theoretical Computer Science 138 (1) (1995) 3–34.

[7] R. Alur, T. A. Henzinger, P. H. Ho, Automatic symbolic verification of embedded systems, IEEE Transactions on Software Engineering 22 (3) (1996) 181–201.

[8] T. A. Henzinger, The theory of hybrid automata, in: M. Inan, R. Kurshan (Eds.), Verification of Digital and Hybrid Systems, Vol. 170 of NATO ASI Series F: Computer and Systems Science, Springer-Verlag, New York, 2000, pp. 265–292.

[9] A. J. van der Schaft, J. M. Schumacher, An Introduction to Hybrid Dynamical Systems, Vol. 251 of Springer Lecture Notes in Control and Information Sciences, Springer, 2000.

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

[11] J. Lygeros, K. H. Johansson, S. Simic, J. Zhang, S. Sastry, Dynamical properties of hybrid automata, IEEE Transactions on Automatic Control 48 (1) (2003) 2–17.

[12] D. A. van Beek, A. Pogromsky, H. Nijmeijer, J. E. Rooda, Convex equations and differential inclusions in hybrid systems, in: 43rd IEEE Conference on Decision and Control, Nassau Bahamas, 2004, pp. 1424–1429.

[13] R. R. H. Schiffelers, D. A. van Beek, K. L. Man, M. A. Reniers, J. E. Rooda, Formal semantics of hybrid Chi, in: K. G. Larsen, P. Niebert (Eds.), Formal Modeling and Analysis of Timed Systems: First International Workshop, FORMATS 2003, Vol. 2791 of Lecture Notes in Computer Science, Springer-Verlag, 2003, pp. 151–165.

[14] T. A. Henzinger, P.-H. Ho, H. Wong-Toi, A user guide to HyTech, in: First International Conference on Tools and Algorithms for the Construction and Analysis of Systems TACAS, Lecture Notes in Computer Science 1019, Springer Verlag, 1995, pp. 41–71.

[15] G. Naumoski, W. Alberts, A discrete-event simulator for systems engineering, Ph.D. thesis, Eindhoven University of Technology (1998).

[16] D. A. van Beek, A. van den Ham, J. E. Rooda, Modelling and control of process industry batch production systems, in: 15th Triennial World Congress of the International Federation of Automatic Control, Barcelona, 2002, CD-ROM.

[17] G. Fábián, A language and simulator for hybrid systems, Ph.D. thesis, Eindhoven University of Technology (1999).

[18] D. A. van Beek, J. E. Rooda, Languages and applications in hybrid modelling and simulation: Positioning of Chi, Control Engineering Practice 8 (1) (2000) 81–91.

[19] V. Bos, J. J. T. Kleijn, Formal specification and analysis of industrial systems, Ph.D. thesis, Eindhoven University of Technology (2002).

[20] V. Bos, J. J. T. Kleijn, Automatic verification of a manufacturing system, Robotics and Computer Integrated Manufacturing 17 (3) (2000) 185–198.

[21] R. R. H. Schiffelers, D. A. van Beek, K. L. Man, M. A. Reniers, J. E. Rooda, A hybrid language for modeling, simulation and verification, in: S. Engell, H. Guéguen, J. Zaytoon (Eds.), IFAC Conference on Analysis and Design of Hybrid Systems, Saint-Malo, Brittany, France, 2003, pp. 235–240.

[22] G. D. Plotkin, A structural approach to operational semantics, Tech. Rep. DIAMI FN-19, Computer Science Department, Aarhus University (1981).

[23] P. J. L. Cuijpers, M. A. Reniers, W. P. M. H. Heemels, Hybrid transition systems, Tech. Rep. CS-Report 02-12, Eindhoven University of Technology, Department of Computer Science, The Netherlands (2002).

[24] J. C. M. Baeten, J. A. Bergstra, Process algebra with propositional signals, Theoretical Computer Science 177 (2) (1997) 381–405.

[25] J. A. Bergstra, C. A. Middelburg, Process algebra for hybrid systems, Theoretical Computer Science 335 (2/3) (2005) 215–280.

[26] F. Breitenecker, I. Husinsky (Eds.), Simulation News Europe, no. 0-40, EUROSIM, 1990-2004, Ch. ARGESIM Comparisons.

[27] M. R. Mousavi, M. A. Reniers, J. F. Groote, Notions of bisimulation and congruence formats for SOS with data, Information and Computation 200 (1) (2005) 107–147.

[28] D. M. R. Park, Concurrency and automata on infinite sequences, in: P. Deussen (Ed.), Proceedings 5^{th} GI Conference, Vol. 104 of LNCS, Springer, 1981, pp. 167–183.

[29] R. Milner, A Calculus of Communicating Systems, Vol. 92 of LNCS, Springer, 1980.

[30] J. C. M. Baeten, C. Verhoef, Concrete process algebra, in: S. Abramsky, D. Gabbay, T. Maibaum (Eds.), Handbook of Logic in Computer Science, Vol. 4 (Semantic Modelling), Oxford University Press, 1995, pp. 149–268.

[31] P. J. L. Cuijpers, M. A. Reniers, Hybrid process algebra, Journal of Logic and Algebraic Programming 62 (2) (2005) 191–245.

[32] W. C. Rounds, H. Song, The ϕ-Calculus: A language for distributed control of reconfigurable embedded systems, in: O. Maler, A. Pnueli (Eds.), Hybrid Systems : Computation and Control, 6th International Workshop, Lecture Notes in Computer Science 2623, Springer-Verlag, 2003, pp. 435–449.

[33] H. Jifeng, From CSP to hybrid systems, in: A. W. Roscoe (Ed.), A Classical Mind, Essays in Honour of C.A.R. Hoare, Prentice Hall, 1994, pp. 171–189.

[34] Z. Chaochen, W. Ji, A. P. Ravn, A formal description of hybrid systems, in: R. Alur, T. A. Henzinger, E. D. Sonntag (Eds.), Hybrid Systems III - Verification and Control, Lecture Notes in Computer Science 1066, Springer-Verlag, 1996, pp. 511–530.

[35] J. J. Vereijken, A process algebra for hybrid systems, in: Bouajjani, Maler (Eds.), The Second European Workshop on Real-Time and Hybrid Systems, Grenoble, France, 1995.

[36] L. Aceto, W. J. Fokkink, C. Verhoef, Structural operational semantics, in: J. Bergstra, A. Ponse, S. Smolka (Eds.), Handbook of Process Algebra, Elsevier, 2001, Ch. 3, pp. 197–292.

[37] T. A. Henzinger, P.-H. Ho, H. Wong-Toi, Algorithmic analysis of nonlinear hybrid systems, IEEE Transactions on Automatic Control 43 (4) (1998) 540–554.

[38] G. Frehse, PHAVer: Algorithmic verification of hybrid systems past HyTech, in: M. Morari, L. Thiele (Eds.), Hybrid Systems: Computation and Control, 8th International Workshop, Vol. 3414 of Lecture Notes in Computer Science, Springer-Verlag, 2005, pp. 258–273.

[39] J. C. M. Baeten, W. P. Weijland, Process Algebra, Vol. 18 of Cambridge Tracts in Theoretical Computer Science, Cambridge University Press, Cambridge, United Kingdom, 1990.

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

[41] D. A. van Beek, K. L. Man, M. A. Reniers, J. E. Rooda, R. R. H. Schiffelers, Syntax and semantics of timed Chi, Tech. Rep. CS-Report 05-09, Eindhoven University of Technology, Department of Computer Science, The Netherlands (2005).

[42] P. van de Brand, M. A. Reniers, P. J. L. Cuijpers, Linearization of hybrid processes, Tech. Rep. CS-Report 04-29, Eindhoven University of Technology, Department of Computer Science, The Netherlands (2004).

[43] R. David, H. Alla, On hybrid Petri nets, Discrete Event Dynamic Systems: Theory & Applications 11 (1-2) (2001) 9–40.

[44] A. D. Febbraro, A. Giua, G. Menga (Eds.), Special Issue on Hybrid Petri Nets, Vol. 11, no. 1 and 2 of Journal of Discrete Event Dynamic Systems, 2001.

[45] R. Alur, T. Dang, J. Esposito, Y. Hur, F. Ivančić, V. Kumar, I. Lee, P. Mishra, G. J. Pappas, O. Sokolsky, Hierarchical modeling and analysis of embedded systems, Proceedings of the IEEE 91 (1) (2003) 11–28.

[46] T. A. Henzinger, Masaccio: A formal model for embedded components, in: First IFIP International Conference on Theoretical Computer Science (TCS), Lecture Notes in Computer Science 1872, Springer-Verlag, 2000, pp. 549–563.

[47] C. A. R. Hoare, Communicating sequential processes, Communications of the ACM 21 (8) (1978) 666–677.

[48] B. I. Silva, K. Richeson, B. H. Krogh, A. Chutinan, Modeling and verification of hybrid dynamical system using CheckMate, in: S. Engell, S. Kowalewski, J. Zaytoon (Eds.), Hybrid Dynamical Systems—Proc. of 4th International Conference on Automation of Mixed Processes, Dortmund, 2000, pp. 323–328.

[49] E. Asarin, O. Bournez, T. Dang, O. Maler., Approximate reachability analysis of piecewise-linear dynamical systems, in: N. A. Lynch, B. H. Krogh (Eds.), Hybrid Systems: Computation and Control, Third International Workshop, Lecture Notes in Computer Science 1790, Springer-Verlag, 2000, pp. 20–31.

[50] R. Alur, T. Dang, F. Ivančić, Progress on reachability analysis of hybrid systems via predicate abstraction, in: O. Maler, A. Pnueli (Eds.), Hybrid Systems: Computation and Control, 6th International Workshop, Lecture Notes in Computer Science 2623, Springer-Verlag, 2003, pp. 4–19.

[51] E. Clarke, A. Fehnker, Z. Han, B. H. Krogh, O. Stursberg, M. Theobald, Verification of hybrid systems based on counterexample-guided abstraction, in: H. Garavel, J. Hatcliff (Eds.), Tools and Algorithms for the Construction and Analysis of Systems, Lecture Notes in Computer Science 2619, Springer-Verlag, 2003, pp. 192–207.

[52] A. Fehnker, E. Clarke, S. K. Jha, B. Krogh, Refining abstractions of hybrid systems using counterexample fragments, in: M. Morari, L. Thiele (Eds.), Hybrid Systems: Computation and Control, 8th International Workshop, Vol. 3414 of Lecture Notes in Computer Science, Springer-Verlag, 2005, pp. 242–257.

[53] S. Ratschan, Z. She, Safety verification of hybrid systems by constraint propagation based abstraction refinement, in: M. Morari, L. Thiele (Eds.), Hybrid Systems: Computation and Control, 8th International Workshop, Vol. 3414 of Lecture Notes in Computer Science, Springer-Verlag, 2005, pp. 573–589.

A Proofs of properties of the Chi semantics

A.1 The proof of Lemma 1

Let p and p' be closed process terms, σ, σ' be valuations, ξ, ξ' be extended valuations, E and E' be environments, a be an action, ρ be a trajectory, and $t \in T$. Then

$$\begin{aligned}
\langle p, \sigma, E\rangle \xrightarrow{\xi, a, \xi'} \langle _, \sigma', E'\rangle \Rightarrow\ & \mathrm{dom}(\sigma) = \mathrm{dom}(\sigma') \wedge \xi_\sigma = \sigma \wedge \xi'_{\sigma'} = \sigma' \\
& \wedge E = E', \\
\langle p, \sigma, E\rangle \xmapsto{t, \rho} \langle p', \sigma', E'\rangle \Rightarrow\ & \mathrm{dom}(\rho) = [0, t] \wedge \rho_\sigma(0) = \sigma \wedge \rho_{\sigma'}(t) = \sigma' \\
& \wedge E = E', \\
\langle p, \sigma, E\rangle \overset{\xi}{\rightsquigarrow} \Rightarrow\ & \xi_\sigma = \sigma,
\end{aligned}$$

where $\langle p, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle \_, \sigma', E'\rangle$ *is an abbreviation for* $\langle p, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle \begin{smallmatrix}\checkmark\\ p'\end{smallmatrix}, \sigma', E'\rangle$ *for some* p'.

PROOF. We prove this lemma by induction on the depth of the proof of the transition in the left-hand-side of the implication and case distinction on the deduction rule applied last in such a proof. The proof for the equality $E = E'$ in the right-hand-side of the implication is trivial, because the equality $E = E'$ holds necessarily according to the result of each χ deduction rule. Therefore, we do not give the proof of this equality for each rule. In what follows, we write E' as E.

Firstly, we give the proofs for $\langle p, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle \_, \sigma', E\rangle \Rightarrow \mathrm{dom}(\sigma) = \mathrm{dom}(\sigma') \wedge \xi_\sigma = \sigma \wedge \xi'_{\sigma'} = \sigma'$. We do not explicitly separate the base cases and the inductive steps.

The rule applied last is

- Rule 1. Then $\xi = \sigma \cup \xi^{\dot{C}L}$ for some $\xi^{\dot{C}L} \in (\dot{C} \cup L) \mapsto \Lambda$ and $\sigma' = \xi'_\sigma$, where ξ'_σ is an abbreviation for $\xi' \restriction \mathrm{dom}(\sigma)$. The domain of the extended valuation ξ' is given by $\mathrm{dom}(\sigma) \cup \dot{C} \cup L$, and the domain of $\xi' \restriction \mathrm{dom}(\sigma)$ is $\mathrm{dom}(\xi') \cap \mathrm{dom}(\sigma)$. Since $\mathrm{dom}(\sigma') = \mathrm{dom}(\xi'_\sigma)$, it is not hard to see that $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma')$. For $\xi = \sigma \cup \xi^{\dot{C}L}$, we obtain $\xi_\sigma = \sigma$. We also have $\sigma' = \xi'_{\sigma'}$, because $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma')$.
- Rules 5 and 6 are similar to the previous case.
- Rule 10. Then, $p = [q]$ for some q and $\langle q, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle \_, \sigma', E\rangle$. By induction we then have $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma') \wedge \xi_\sigma = \sigma \wedge \xi'_{\sigma'} = \sigma'$.
- Rule 13. Then $p \equiv u \curvearrowright q$ for some u and q, $\langle q, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle \_, \sigma', E\rangle$ and $\xi \models u$. By induction we then have $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma') \wedge \xi_\sigma = \sigma \wedge \xi'_{\sigma'} = \sigma'$.
- Rule 16. Then $p \equiv q_1 ; q_2$ for some q_1 and q_2, $\langle q_1, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle \_, \sigma', E\rangle$ and $\langle q_2, \sigma', E\rangle \overset{\xi'}{\rightsquigarrow}$. By induction we then have $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma') \wedge \xi_\sigma = \sigma \wedge \xi'_{\sigma'} = \sigma'$.
- Rule 17. Then $p \equiv q_1 ; q_2$ for some q_1 and q_2, and $\langle q_1, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle \_, \sigma', E\rangle$. By induction we then have $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma') \wedge \xi_\sigma = \sigma \wedge \xi'_{\sigma'} = \sigma'$.
- Rule 20. Then $p \equiv b \to q$ for some b and q, $\langle q, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle \_, \sigma', E\rangle$ and $\xi \models b$. By induction we then have $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma') \wedge \xi_\sigma = \sigma \wedge \xi'_{\sigma'} = \sigma'$.
- Rule 25. Then $p \equiv q_1 \mathbin{[\![} q_2$ for some q_1 and q_2, and $\langle q_1, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle \_, \sigma', E\rangle$ and $\langle q_2, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$. By induction we then have $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma') \wedge \xi_\sigma = \sigma \wedge \xi'_{\sigma'} = \sigma'$.
- Rule 28. Then $p \equiv q_1 \parallel q_2$ for some q_1 and q_2, and $\langle q_1, \sigma, E_a\rangle \xrightarrow{\xi,a,\xi'} \langle \_, \sigma', E_a\rangle$ and $\langle q_2, \sigma, E_b\rangle \xrightarrow{\xi,b,\xi'} \langle \_, \sigma', E_b\rangle$ for some (unimportant) actions a and b, and

some (unimportant) environments E_a and E_b. By induction we then have $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma') \wedge \xi_\sigma = \sigma \wedge \xi'_{\sigma'} = \sigma'$.

- Rule 29. Then $p \equiv q_1 \parallel q_2$ for some q_1 and q_2, and $\langle q_1, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$, $\langle q_1, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle \_, \sigma', E\rangle$ and $\langle q_2, \sigma', E\rangle \overset{\xi'}{\rightsquigarrow}$. By induction we then have $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma') \wedge \xi_\sigma = \sigma \wedge \xi'_{\sigma'} = \sigma'$.
- Rule 32. Then $p \equiv \partial_A(q)$ for some A and q, $\langle q, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle \_, \sigma', E\rangle$, and $a \notin A$. By induction we then have $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma') \wedge \xi_\sigma = \sigma \wedge \xi'_{\sigma'} = \sigma'$.
- Rule 35. Then $p \equiv \upsilon_H(q)$ for some $\mathcal{H}$ and q, and $\langle q, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle \_, \sigma', E\rangle$. By induction we then have $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma') \wedge \xi_\sigma = \sigma \wedge \xi'_{\sigma'} = \sigma'$.
- Rule 38. Then $p \equiv X$ for some X, $E = (C, J, L, H, R)$ and $\langle R(X), \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle \_, \sigma', E\rangle$. By induction, we have $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma') \wedge \xi_\sigma = \sigma \wedge \xi'_{\sigma'} = \sigma'$.
- Rule 41. Then $E = (C, J, L, H, R)$ and $p \equiv \iota_{J^+}(q)$ for some J^+ and q, and $(C, J \cup J^+, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \_, \sigma', E\rangle$. By induction we then have $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma') \wedge \xi_\sigma = \sigma \wedge \xi'_{\sigma'} = \sigma'$.
- Rule 44. We assume $\langle p, \sigma, E\rangle \xrightarrow{\xi_x,a,\xi_y} \langle \_, \sigma', E\rangle$ for some ξ_x and ξ_y. Then, we have $E = (C, J, L, H, R), p \equiv [\![_{\mathrm{V}}\, \sigma_{\mathrm{dx}_\perp}, \{\mathbf{x}\}, \{\mathbf{g}\} \mid q\,]\!]$ for some q, $\sigma_{\mathrm{dx}_\perp}$, $\mathbf{x}$, $\mathbf{g}$, $(C \cup \{\mathbf{x}'\}, J, L \cup \{\mathbf{g}'\}, H, R) \Vdash \langle q[\mathbf{d}', \mathbf{x}', \mathbf{g}'/\mathbf{d}, \mathbf{x}, \mathbf{g}], \sigma \cup \sigma_{\mathrm{d}'\mathrm{x}'}\rangle \xrightarrow{\xi,a,\xi'} \langle \_, \sigma''\rangle$ for some $\mathbf{d}, \mathbf{d}', \mathbf{x}', \mathbf{g}', \sigma_{\mathrm{d}'\mathrm{x}'}, \sigma''$, $\sigma' = \sigma''_\sigma; \xi, \xi'$such that $\xi_x = \xi \upharpoonright (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)$ and $\xi_y = \xi' \upharpoonright (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)$. Note that the syntactical equality of p' is not given, because it is irrelevant for this proof.
  - Firstly, we have to show that $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma''_\sigma)$. By induction, we know that $\mathrm{dom}(\sigma \cup \sigma_{\mathrm{d}'\mathrm{x}'}) = \mathrm{dom}(\sigma) \cup \mathrm{dom}(\sigma_{\mathrm{d}'\mathrm{x}'}) = \mathrm{dom}(\sigma'')$. On the other hand, $\mathrm{dom}(\sigma''_\sigma) = \mathrm{dom}(\sigma'') \cap \mathrm{dom}(\sigma) = (\mathrm{dom}(\sigma) \cup \mathrm{dom}(\sigma_{\mathrm{d}'\mathrm{x}'})) \cap \mathrm{dom}(\sigma) = \mathrm{dom}(\sigma)$, i.e. $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma''_\sigma)$.
  - Secondly, we have to show that $\xi_x \upharpoonright \mathrm{dom}(\sigma) = \sigma$. By induction, we know that $\xi \upharpoonright \mathrm{dom}(\sigma \cup \sigma_{\mathrm{d}'\mathrm{x}'}) = \sigma \cup \sigma_{\mathrm{d}'\mathrm{x}'}$, then $\xi \upharpoonright \mathrm{dom}(\sigma) = \sigma$ and $\xi \upharpoonright \mathrm{dom}(\sigma_{\mathrm{d}'\mathrm{x}'}) = \sigma_{\mathrm{d}'\mathrm{x}'}$. On the other hand, $\xi_x \upharpoonright \mathrm{dom}(\sigma) = (\xi \upharpoonright (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)) \upharpoonright \mathrm{dom}(\sigma) = \xi \upharpoonright \mathrm{dom}(\sigma) = \sigma$, i.e. $\xi_x \upharpoonright \mathrm{dom}(\sigma) = \sigma$.
  - Thirdly, we have to show that $\xi_y \upharpoonright \mathrm{dom}(\sigma''_\sigma) = \sigma''_\sigma$. By induction, we know that $\xi \upharpoonright \mathrm{dom}(\sigma'') = \sigma''$. On the other hand, $\xi_y \upharpoonright \mathrm{dom}(\sigma''_\sigma) = (\xi' \upharpoonright (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)) \upharpoonright \mathrm{dom}(\sigma''_\sigma) = (\xi' \upharpoonright (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)) \upharpoonright (\mathrm{dom}(\sigma'') \cap \mathrm{dom}(\sigma)) = \xi' \upharpoonright (\mathrm{dom}(\sigma'') \cap \mathrm{dom}(\sigma))$. From $\sigma'' = \xi \upharpoonright \mathrm{dom}(\sigma'')$, we obtain $\sigma''_\sigma = \sigma'' \upharpoonright \mathrm{dom}(\sigma) = (\xi \upharpoonright \mathrm{dom}(\sigma'')) \upharpoonright \mathrm{dom}(\sigma)$. It is not hard to see that $\xi' \upharpoonright (\mathrm{dom}(\sigma'') \cap \mathrm{dom}(\sigma)) = (\xi \upharpoonright \mathrm{dom}(\sigma'')) \upharpoonright \mathrm{dom}(\sigma)$, which also means $\xi_y \upharpoonright \mathrm{dom}(\sigma''_\sigma) = \sigma''_\sigma$.
- Rules 47, 48 and 51. The proofs are similar. We only give the proof for Rule 47. Then $p \equiv [\![_{\mathrm{H}}\, \{\mathbf{h}\} \mid q\,]\!]$ for some $\mathbf{h}$, q, $E = (C, J, L, H, R)$, $(C, J, L, H \cup \{\mathbf{h}'\}, R) \Vdash \langle q[\mathbf{h}'/\mathbf{h}], \sigma\rangle \xrightarrow{\xi,b,\xi'} \langle \_, \sigma', E\rangle$ for some unimportant action b for this proof, $\mathbf{h}'$ and $h \in \{\mathbf{h}'\}$ for some h. By induction we then have $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma') \wedge \xi_\sigma = \sigma \wedge \xi'_{\sigma'} = \sigma'$.

The rules that have not been considered could not have been applied last since they conclude a time transition or a consistency predicate.

Secondly, we give the proofs for $\langle p, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle p', \sigma', E\rangle \Rightarrow \mathrm{dom}(\rho) = [0,t] \wedge \rho_\sigma(0) = \sigma \wedge \rho_{\sigma'}(t) = \sigma'$. We do not explicitly separate the base cases and the inductive steps.

The rule applied last is

- Rule 3. Then, $p \equiv u \equiv p'$ for some u, $E = (C, J, L, H, R)$, $\rho \in \Omega_{FG}(\sigma, C, L, u, t)$, and $\sigma' = \rho_\sigma(t)$. Then, by the definition of Ω_{FG}, $\mathrm{dom}(\rho) = [0,t]$, and $\rho(0) \restriction \mathrm{dom}(\sigma) = \rho_\sigma(0) = \sigma$ necessarily. From $\sigma' = \rho_\sigma(t)$, we know that $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma')$. Therefore, we also have $\sigma' = \rho_{\sigma'}(t)$.
- Rule 11. Then $p \equiv [q] \equiv p'$ for some q, $\rho \in \Omega_{\sigma E t}$ and $\sigma' = \rho_\sigma(t)$. Then, by the definition of Ω_{FG}, $\mathrm{dom}(\rho) = [0,t]$, and $\rho(0) \restriction \mathrm{dom}(\sigma) = \rho_\sigma(0) = \sigma$ necessarily. From $\sigma' = \rho_\sigma(t)$, we know that $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma')$. Therefore, we have also $\sigma' = \rho_{\sigma'}(t)$.
- Rule 14. Then $p \equiv u \curvearrowright q$ for some u and q, $\langle q, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle p', \sigma', E\rangle$ and $\rho(0) \models u$. By induction we then have $\mathrm{dom}(\rho) = [0,t] \wedge \rho_\sigma(0) = \sigma \wedge \rho_{\sigma'}(t) = \sigma'$.
- Rule 18. Then $p \equiv q_1 ; q_2$ for some q_1 and q_2, $\langle q_1, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q_1', \sigma', E\rangle$ for some q_1' and $p' \equiv q_1' ; q_2$. By induction we then have $\mathrm{dom}(\rho) = [0,t] \wedge \rho_\sigma(0) = \sigma \wedge \rho_{\sigma'}(t) = \sigma'$.
- Rule 21. Then $p \equiv b \to q$ for some b and q, $\langle q, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q', \sigma', E\rangle$ for some q' such that $p' \equiv b \to q'$, and $\forall_{s \in [0,t]}\, \rho(s) \models b$. By induction we then have $\mathrm{dom}(\rho) = [0,t] \wedge \rho_\sigma(0) = \sigma \wedge \rho_{\sigma'}(t) = \sigma'$.
- Rule 22. Then $p \equiv b \to q \equiv p'$ for some b and q, $\rho \in \Omega_{\sigma E t}$ and $\sigma' = \rho_\sigma(t)$ (some irrelevant information for the proof is omitted). By the definition of Ω_{FG}, $\mathrm{dom}(\rho) = [0,t]$, and $\rho(0) \restriction \mathrm{dom}(\sigma) = \rho_\sigma(0) = \sigma$ necessarily. From $\sigma' = \rho_\sigma(t)$, we know that $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma')$. Therefore, we have also $\sigma' = \rho_{\sigma'}(t)$.
- Rule 26. Then $p \equiv q_1 \mathbin{[\!]} q_2$ for some q_1 and q_2, $\langle q_1, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q_1', \sigma', E\rangle$ and $\langle q_2, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q_2', \sigma', E\rangle$ for some q_1' and q_2', and $p' \equiv q_1' \mathbin{[\!]} q_2'$. By induction we then have $\mathrm{dom}(\rho) = [0,t] \wedge \rho_\sigma(0) = \sigma \wedge \rho_{\sigma'}(t) = \sigma'$.
- Rule 30. Then $p \equiv q_1 \parallel q_2$ for some q_1 and q_2, $\langle q_1, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q_1', \sigma', E\rangle$ and $\langle q_2, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q_2', \sigma', E\rangle$, for some q_1' and q_2', and $p' \equiv q_1' \parallel q_2'$. By induction we then have $\mathrm{dom}(\rho) = [0,t] \wedge \rho_\sigma(0) = \sigma \wedge \rho_{\sigma'}(t) = \sigma'$.
- Rule 33. Then $p \equiv \partial_A(q)$ for some A and q, $\langle q, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q', \sigma, E\rangle$ for some q', and $p' \equiv \partial_A(q')$. By induction we then have $\mathrm{dom}(\rho) = [0,t] \wedge \rho_\sigma(0) = \sigma \wedge \rho_{\sigma'}(t) = \sigma'$.
- Rule 37. Then $p \equiv \upsilon_H(q)$ for some H and q, and $\langle q, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q', \sigma, E\rangle$ for some q', and $p' \equiv \upsilon_H(q')$ (some irrelevant information for this proof is omitted). By induction we then have $\mathrm{dom}(\rho) = [0,t] \wedge \rho_\sigma(0) = \sigma \wedge \rho_{\sigma'}(t) = \sigma'$.
- Rule 39. Then $p \equiv X$ for some X, $E = (C, J, L, H, R)$ and $\langle R(X), \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle p', \sigma', E\rangle$. By induction we then have $\mathrm{dom}(\rho) = [0,t] \wedge \rho_\sigma(0) = \sigma \wedge \rho_{\sigma'}(t) = \sigma'$.
- Rule 42. Then $p \equiv \iota_{J^+}(q)$ for some term q and set J^+, $E = (C, J, L, H, R)$, $(C, J \cup J^+, L, H, R) \Vdash \langle q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle q', \sigma'\rangle$ for some q', and $p' \equiv \iota_{J^+}(q')$. By induction we then have $\mathrm{dom}(\rho) = [0,t] \wedge \rho_\sigma(0) = \sigma \wedge \rho_{\sigma'}(t) = \sigma'$.

- Rule 45. We assume $\langle p, \sigma, E\rangle \xmapsto{t,\rho'} \langle p', \sigma', E\rangle$ for some ρ'. Then $E = C,J \cup J^+,L, H, R$, $p \equiv [\![_{\mathrm{V}}\, \sigma_{\mathrm{dx}_\perp}, \{\mathbf{x}\}, \{\mathbf{g}\} \mid q\,]\!]$ for some q, $\sigma_{\mathrm{dx}_\perp}$,$\mathbf{x}$,$\mathbf{g}$, $(C \cup \{\mathbf{x}'\},J, L \cup \{\mathbf{g}'\},H, R) \vdash \langle q[\mathbf{d}', \mathbf{x}', \mathbf{g}'/\mathbf{d}, \mathbf{x}, \mathbf{g}], \sigma \cup \sigma_{\mathrm{d}'\mathrm{x}'}\rangle \xmapsto{t,\rho} \langle q', \sigma''\rangle$ for some q', $\mathbf{d}$, $\mathbf{d}'$, $\mathbf{x}'$, $\mathbf{g}'$, $\sigma_{\mathrm{d}'\mathrm{x}'}$, σ'', $\sigma' = \sigma''_\sigma$, and $\rho' = \rho_{\sigma\dot{C}L} = \rho \downarrow (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)$. Note that the syntactical equality of p' is not given, because it is irrelevant for this proof.
  - Firstly, we have to show that $\mathrm{dom}(\rho \downarrow (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)) = [0, t]$. By induction we know that $\mathrm{dom}(\rho) = [0, t]$. On the other hand, we have $\mathrm{dom}(\rho \downarrow (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)) = \mathrm{dom}(\rho) = [0, t]$.
  - Secondly, we have to show that $\rho' \downarrow \mathrm{dom}(\sigma)(0) = (\rho \downarrow (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)) \downarrow \mathrm{dom}(\sigma)(0) = \sigma$. By induction we know that $\rho \downarrow (\mathrm{dom}(\sigma \cup \sigma_{\mathrm{d}'\mathrm{x}'}))(0) = \sigma \cup \sigma_{\mathrm{d}'\mathrm{x}'}$. Then, we have also $\rho \downarrow \mathrm{dom}(\sigma)(0) = \sigma$ and $\rho \downarrow \mathrm{dom}(\sigma_{\mathrm{d}'\mathrm{x}'})(0) = \sigma_{\mathrm{d}'\mathrm{x}'}$. On the other hand, $\rho' \downarrow \mathrm{dom}(\sigma)(0) = (\rho \downarrow (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)) \downarrow \mathrm{dom}(\sigma)(0) = \rho \downarrow \mathrm{dom}(\sigma)(0) = \sigma$.
  - Thirdly, we have to show that $\rho' \downarrow \mathrm{dom}(\sigma''_\sigma)(t) = (\rho \downarrow (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)) \downarrow \mathrm{dom}(\sigma''_\sigma)(t) = \sigma''_\sigma$. By induction we know that $\rho \downarrow \mathrm{dom}(\sigma'')(t) = \sigma''$. Then, we have $(\rho \downarrow \mathrm{dom}(\sigma'')) \downarrow \mathrm{dom}(\sigma)(t) = \sigma'' \downarrow \mathrm{dom}(\sigma) = \sigma''_\sigma$. On the other hand, $\rho' \downarrow \mathrm{dom}(\sigma''_\sigma)(t) = ((\rho \downarrow (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)) \downarrow \mathrm{dom}(\sigma''_\sigma)(t) = ((\rho \downarrow (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)) \downarrow (\mathrm{dom}(\sigma'') \cap \mathrm{dom}(\sigma)))(t) = \rho \downarrow (\mathrm{dom}(\sigma'') \cap \mathrm{dom}(\sigma))(t) = (\rho \downarrow \mathrm{dom}(\sigma'')) \downarrow \mathrm{dom}(\sigma)(t) = \sigma''_\sigma$.
- Rules 49 and 52. The proofs are similar. We only give the proof for Rule 49. Then $p \equiv [\![_{\mathrm{H}}\, \{\mathbf{h}\} \mid q\,]\!]$ for some $\mathbf{h}$, q, $E = (C, J, L, H, R)$, $(C,\ J,\ L, H \cup \{\mathbf{h}'\}, R) \Vdash \langle q[\mathbf{h}'/\mathbf{h}], \sigma\rangle \xmapsto{t,\rho} \langle q', \sigma'\rangle$ for some q'. Note that the syntactical equality of p' is not given, because it is irrelevant for this proof. By induction we then have $\mathrm{dom}(\rho) = [0, t] \wedge \rho_\sigma(0) = \sigma \wedge \rho_{\sigma'}(t) = \sigma'$.

The rules that have not been considered could not have been applied last since they conclude an action transition or a consistency predicate.

The proof for $\langle p, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}\!\Rightarrow \xi_\sigma = \sigma$ is trivial. According to all χ deduction rules for consistency predicates, $\xi = \sigma \cup \xi^{\dot{C}L}$ for some $\xi^{\dot{C}L} \in (\dot{C} \cup L) \mapsto \Lambda$ necessarily. Then we have $\xi_\sigma = \sigma$.

A.2 The proof of Lemma 2

Let p and p' be closed process terms, σ and σ' be valuations, E and E' be environments, ξ and ξ' be extended valuations and a be an action. Then

$$\langle p, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \implies \langle p, \sigma, E\rangle \overset{\xi}{\rightsquigarrow},$$

where $\langle p, \sigma, E\rangle \xrightarrow{\xi,a,\xi'}$ is an abbreviation for $\langle p, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle \overset{\checkmark}{p'}, \sigma', E'\rangle$ for some p', σ', and E'.

PROOF. We prove this lemma by induction on the depth of the proof of $\langle p, \sigma, E\rangle \xrightarrow{\xi,a,\xi'}$ using case distinction based on the deduction rule applied last. We do not explicitly separate the base cases and the inductive steps.

The rule applied last is

- Rule 1. Then $p \equiv W : r \gg l_{\mathrm{a}}$ for some W, r, l_{a}, $\xi = \sigma \cup \xi^{\dot{C}L}$ for some $\xi^{\dot{C}L} \in (\dot{C} \cup L) \mapsto \Lambda$, and $a = l_{\mathrm{a}}$. Therefore, by Rule 2, we have $\langle W : r \gg l_{\mathrm{a}}, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$.
- Rule 5. Then $p \equiv h \,!!\, \mathbf{e}_n$ for some h and $\mathbf{e}_n$, $\xi = \sigma \cup \xi^{\dot{C}L}$ for some $\xi^{\dot{C}L} \in (\dot{C} \cup L) \mapsto \Lambda$, and $a = \mathrm{isa}(h, [\xi(\mathbf{e}_n)])$. Therefore, by Rule 7, we have $\langle h \,!!\, \mathbf{e}_n, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$.
- Rule 6. Then $p \equiv h \,??\, \mathbf{x}_n$ for some h and $\mathbf{x}_n$, $\xi = \sigma \cup \xi^{\dot{C}L}$ for some $\xi^{\dot{C}L} \in (\dot{C} \cup L) \mapsto \Lambda$, and $a = \mathrm{ira}(h, [\mathbf{c}_n], \{\mathbf{x}_n\})$ for some $\mathbf{c}_n$. Then, by Rule 8, we have $\langle h \,??\, \mathbf{x}_n, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$.
- Rule 10. Then, $p = [q]$ for some q, $E = (C, J, L, H, R)$ and $\langle q, \sigma, E\rangle \xrightarrow{\xi,a,\xi'}$. By induction we then have $\langle q, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$. Then, by Rule 12, we have $\langle [q], \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$, and $\xi = \sigma \cup \xi^{\dot{C}L}$ for some $\xi^{\dot{C}L} \in (\dot{C} \cup L) \mapsto \Lambda$.
- Rule 13. Then $p \equiv u \curvearrowright q$ for some u and q, $\langle q, \sigma, E\rangle \xrightarrow{\xi,a,\xi'}$ and $\xi \models u$. By induction $\langle q, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$. Then, by Rule 15, we have $\langle u \curvearrowright q, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$.
- Rule 16. Then $p \equiv q_1;\ q_2$ for some q_1 and q_2, $\langle q_1, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma', E'\rangle$ and $\langle q_2, \sigma', E\rangle \overset{\xi'}{\rightsquigarrow}$. By induction $\langle q_1, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$. Then, by Rule 19, we have $\langle q_1;\ q_2, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$.
- Rule 17. Then $p \equiv q_1;\ q_2$ for some q_1 and q_2, and $\langle q_1, \sigma, E\rangle \xrightarrow{\xi,a,\xi'} \langle q_1', \sigma', E'\rangle$ for some q_1'. By induction $\langle q_1, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$. Then, by Rule 19, we have $\langle q_1;\ q_2, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$.
- Rule 20. Then $p \equiv b \to q$ for some b and q, $\langle q, \sigma, E\rangle \xrightarrow{\xi,a,\xi'}$ and $\xi \models b$. By induction $\langle q, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$. Then, by Rule 23, we have $\langle b \to q, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$.
- Rule 25. Then $p \equiv q_1 \,[\!]\, q_2$ for some q_1 and q_2, and $\langle q_1, \sigma, E\rangle \xrightarrow{\xi,a,\xi'}$ and $\langle q_2, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$. By induction $\langle q_1, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$. Then, by Rule 27, we have $\langle q_1 \,[\!]\, q_2, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$.
- Rule 28. Then $p \equiv q_1 \parallel q_2$ for some q_1 and q_2, and $\langle q_1, \sigma, E_a\rangle \xrightarrow{\xi,a,\xi'}$ and $\langle q_2, \sigma, E_b\rangle \xrightarrow{\xi,b,\xi'}$ for some (unimportant) actions a and b, and some (unimportant) environments E_a and E_b. By induction $\langle q_1, \sigma, E_a\rangle \overset{\xi}{\rightsquigarrow}$ and $\langle q_2, \sigma, E_b\rangle \overset{\xi}{\rightsquigarrow}$. Then, by Rule 31 and by Lemma 4, we have $\langle q_1 \,[\!]\, q_2, \sigma, E_a\rangle \overset{\xi}{\rightsquigarrow}$.
- Rule 29. Then $p \equiv q_1 \parallel q_2$ for some q_1 and q_2, and $\langle q_1, \sigma, E\rangle \xrightarrow{\xi,a,\xi'}$ and $\langle q_2, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$. By induction $\langle q_1, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$. Then, by Rule 31, we have $\langle q_1 \,[\!]\, q_2, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$.
- Rule 32. Then $p \equiv \partial_A(q)$ for some A and q, $\langle q, \sigma, E\rangle \xrightarrow{\xi,a,\xi'}$, and $a \notin A$. By induction we then have $\langle q, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 34, we obtain $\langle \partial_A(q), \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$.
- Rule 35. Then $p \equiv \upsilon_H(q)$ for some H and q, and $\langle q, \sigma, E\rangle \xrightarrow{\xi,a,\xi'}$. By induction

we then have $\langle q, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 36, we obtain $\langle \upsilon_H(q), \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$.

- Rule 38. Then $p \equiv X$ for some X and $E = (C, J, L, H, R)$ and $\langle R(X),\sigma,E \rangle \xrightarrow{\xi,a,\xi'}$. By induction, we have $\langle R(X), \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$. Then, by Rule 40, $\langle X, \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$.
- Rule 41. Then $E = (C, J, L, H, R)$ and $p \equiv \iota_{J^+}(q)$ for some J^+ and q, and $(C, J \cup J^+, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi,a,\xi'}$. By induction we have $(C, J \cup J^+, L, H, R) \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. By Rule 43, we have $\langle \iota_{J^+}(q), \sigma, E\rangle \overset{\xi}{\rightsquigarrow}$.
- Rule 44. We assume $\langle p, \sigma, E\rangle \xrightarrow{\xi_x,a,\xi_y}$ for some ξ_x, and ξ_y. Then, we have that $E = (C, J, L, H, R)$, $p \equiv [\![_{\mathrm{V}}\, \sigma_{\mathrm{dx}_\perp}, \{\mathbf{x}\}, \{\mathbf{g}\} \mid q\,]\!]$ for some q, $\sigma_{\mathrm{dx}_\perp}$, $\mathbf{x}$, $\mathbf{g}$, and $(C \cup \{\mathbf{x}'\}, J, L \cup \{\mathbf{g}'\}, H, R) \Vdash \langle q[\mathbf{d}', \mathbf{x}', \mathbf{g}'/\mathbf{d}, \mathbf{x}, \mathbf{g}], \sigma \cup \sigma_{\mathrm{d}'\mathrm{x}'}\rangle \xrightarrow{\xi,a,\xi'}$ for some $\mathbf{d}$, $\mathbf{d}'$, $\mathbf{x}'$, $\mathbf{g}'$, $\sigma_{\mathrm{d}'\mathrm{x}'}$, ξ, ξ' such that $\xi_x = \xi \restriction (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)$ and $\xi_y = \xi' \restriction (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)$. By induction we have $(C \cup \{\mathbf{x}'\}, J, L \cup \{\mathbf{g}'\}, H, R) \Vdash \langle q[\mathbf{d}', \mathbf{x}', \mathbf{g}'/\mathbf{d}, \mathbf{x}, \mathbf{g}], \sigma \cup \sigma_{\mathrm{d}'\mathrm{x}'}\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 46, we obtain $(C, J, L, H, R) \Vdash \langle [\![_{\mathrm{V}}\, \sigma_{\mathrm{dx}_\perp}, \{\mathbf{x}\}, \{\mathbf{g}\} \mid p\,]\!], \sigma\rangle \overset{\xi\restriction(\mathrm{dom}(\sigma)\cup\dot{C}\cup L)}{\rightsquigarrow}$.
- Rules 47, 48 and 51. The proofs are similar. We only give the proof for Rule 47. Then $p \equiv [\![_{\mathrm{H}}\, \{\mathbf{h}\} \mid q\,]\!]$ for some $\mathbf{h}$, q, $E = (C, J, L, H, R)$, $\langle q[\mathbf{h}'/\mathbf{h}], \sigma, (C,J,L,H \cup \{\mathbf{h}'\},R)\rangle \xrightarrow{\xi,b,\xi'}$ for some unimportant action b for this proof, $\mathbf{h}'$ and $h \in \{\mathbf{h}'\}$ for some h. By induction we then $\langle q[\mathbf{h}'/\mathbf{h}], \sigma,(C,J,L,H \cup \{\mathbf{h}'\},R)\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 50, we obtain $(C, J, L, H, R) \Vdash \langle [\![_{\mathrm{H}}\, \{\mathbf{h}\} \mid q\,]\!], \sigma\rangle \overset{\xi}{\rightsquigarrow}$.

The rules that have not been considered could not have been applied last since they conclude a time transition or a consistency predicate.

A.3 The proof of Lemma 3

Let p and p' be closed process terms, σ and σ' be valuations, E and E' be environments, $t \in T$, and ρ be a trajectory. Then,

$$\langle p, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle p', \sigma', E'\rangle \Rightarrow \langle p, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}.$$

PROOF. We prove this lemma by induction on the depth of the proof of $\langle p, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle p', \sigma', E'\rangle$ using case distinction based on the deduction rule applied last. We do not explicitly separate the base cases and the inductive steps.

The rule applied last is

- Rule 3. Then, $p \equiv u \equiv p'$ for some u, $E = (C, J, L, H, R)$, $\rho \in \Omega_{FG}(\sigma, C, L, u, t)$. Then, by definition, $\rho(0) \models u$ and $\rho(0) \restriction \mathrm{dom}(\sigma) = \sigma$. Thus $\rho(0) = \sigma \cup \xi^{\dot{C}L}$

for some $\xi^{\dot{C}L} \in (\dot{C} \cup L) \mapsto \Lambda$. Therefore, by Rule 4, we have $\langle u, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$.

- Rule 11. Then $p \equiv [q]$ for some q and $\rho(0) \in \Omega_{\sigma E t}$. Then, by definition, $\rho(0) \restriction \mathrm{dom}(\sigma) = \sigma$. Thus $\rho(0) = \sigma \cup \xi^{\dot{C}L}$ for some $\xi^{\dot{C}L} \in (\dot{C} \cup L) \mapsto \Lambda$. Therefore, by Rule 12, $\langle [q], \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$.
- Rule 14. Then $p \equiv u \curvearrowright q$ for some u and p, $\langle q, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle p', \sigma', E'\rangle$ and $\rho(0) \models u$. Therefore, by induction, $\langle q, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$. Then, by Rule 15, $\langle u \curvearrowright q, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$.
- Rule 18. Then $p \equiv q_1;\ q_2$ for some q_1 and q_2, $\langle q_1, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q_1', \sigma', E'\rangle$ for some q_1', and $p' \equiv q_1';\ q_2$. By induction we have $\langle q_1, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$, and thus by application of Rule 19 we have $\langle q_1;\ q_2, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$.
- Rule 21. Then $p \equiv b \to q$ for some b and q, $\langle q, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q', \sigma', E'\rangle$ for some q' such that $p' \equiv b \to q'$, and $\forall_{s\in[0,t]}\, \rho(s) \models b$. By induction we have $\langle q, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$. Since we also have $\rho(0) \models b$, we have, by Rule 23, $\langle b \to q, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$.
- Rule 22. Then $p \equiv b \to q$ for some b and q, $\rho \in \Omega_{\sigma E t}$, $\forall_{s\in(0,t)}\, \rho(s) \models \neg b$, $\rho(0) \models b \implies \langle q, \sigma, E\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle q', \sigma'', E''\rangle$ for some q', σ'' and E''. In case $\rho(0) \models \neg b$, we also have $\sigma \cup \xi^{\dot{C}L} \models \neg b$ for some $\xi^{\dot{C}L} \in (\dot{C} \cup L) \mapsto \Lambda$. Then, by Rule 24, $\langle b \to q, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$. In case $\rho(0) \models b$, we have $\langle q, \sigma, E\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle q', \sigma'', E''\rangle$. By induction we then have $\langle q, \sigma, E\rangle \overset{\rho\restriction\{0\}(0)}{\rightsquigarrow}$, which gives $\langle q, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$. By Rule 23 we then have $\langle b \to q, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$.
- Rule 26. Then $p \equiv q_1 \mathbin{[\!]} q_2$ for some q_1 and q_2, $\langle q_1, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q_1', \sigma', E'\rangle$ for some q_1', $\langle q_2, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q_2', \sigma', E'\rangle$ for some q_2', and $p' \equiv q_1' \mathbin{[\!]} q_2'$. By induction we have $\langle q_1, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$ and $\langle q_2, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$, and thus by application of Rule 27 we have $\langle q_1 \mathbin{[\!]} q_2, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$.
- Rule 30. Then $p \equiv q_1 \parallel q_2$ for some q_1 and q_2, $\langle q_1, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q_1', \sigma', E'\rangle$ for some q_1', $\langle q_2, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q_2', \sigma', E'\rangle$ for some q_2', and $p' \equiv q_1' \parallel q_2'$. By induction we have $\langle q_1, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$ and $\langle q_2, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$, and thus by application of Rule 31 we have $\langle q_1 \parallel q_2, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$.
- Rule 33. Then $p \equiv \partial_A(q)$ for some A and q, $\langle q, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q', \sigma, E\rangle$ for some q', and $p' \equiv \partial_A(q')$. By induction we then have $\langle q, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$. By Rule 34, we obtain $\langle \partial_A(q), \sigma\rangle \overset{\rho(0)}{\rightsquigarrow}$.
- Rule 37. Then $p \equiv \upsilon_H(q)$ for some H and q, and $\langle q, \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle q', \sigma, E\rangle$ for some q', and $p' \equiv \upsilon_H(q')$ (some irrelevant information for the proof is omitted). By induction we then have $\langle q, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$. By Rule 36, we obtain $\langle \upsilon_H(q), \sigma\rangle \overset{\rho(0)}{\rightsquigarrow}$.
- Rule 39. Then $p \equiv X$ for some X, $E = (C, J, L, H, R)$ and $\langle R(X), \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle p', \sigma', E'\rangle$. As the proof for $\langle R(X), \sigma, E\rangle \overset{t,\rho}{\longmapsto} \langle p', \sigma', E'\rangle$ has smaller depth, by

induction we have $\langle R(X), \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$. Then, by Rule 40, we have $\langle X, \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$ as well.

- Rule 42. Then $p \equiv \iota_{J^+}(q)$ for some term q and set J^+, $E = (C, J, L, H, R)$, $(C, J \cup J^+, L, H, R) \Vdash \langle q, \sigma, \rangle \overset{t,\rho}{\longmapsto} \langle q', \sigma'\rangle$ for some q', and $p' \equiv \iota_{J^+}(q')$. By induction we then have $(C, J \cup J^+, L, H, R) \Vdash \langle q, \sigma\rangle \overset{\rho(0)}{\rightsquigarrow}$. From Rule 43, we deduce $\langle \iota_{J^+}(q), \sigma, E\rangle \overset{\rho(0)}{\rightsquigarrow}$.
- Rule 45. Then $p \equiv [\![_{\mathrm{V}}\, \sigma_{\mathrm{dx}_\perp}, \{\mathbf{x}\}, \{\mathbf{g}\} \mid q\,]\!]$ for some q, $E = (C,J,L,H,R)$, $\sigma_{\mathrm{dx}_\perp}$, $\mathbf{x}$, $\mathbf{g}$, $(C \cup \{\mathbf{x}'\}, J, L \cup \{\mathbf{g}'\}, H, R) \Vdash \langle q[\mathbf{d}', \mathbf{x}', \mathbf{g}'/\mathbf{d}, \mathbf{x}, \mathbf{g}], \sigma \cup \sigma_{\mathrm{d}'\mathrm{x}'}\rangle \overset{t,\rho}{\longmapsto} \langle q', \sigma''\rangle$ for some ρ, q', $\mathbf{d}$, $\mathbf{d}'$, $\mathbf{x}'$, $\mathbf{g}'$, $\sigma_{\mathrm{d}'\mathrm{x}'}$, σ'', $\sigma' = \sigma''_\sigma$, and $\rho' = \rho_{\sigma\dot{C}L} = \rho \downarrow (\mathrm{dom}(\sigma) \cup \dot{C} \cup L)$. Note that the syntactical equality of p' is not given, because it is irrelevant for this proof. By induction we then have $(C \cup \{\mathbf{x}'\}, J, L \cup \{\mathbf{g}'\}, H, R) \Vdash \langle q[\mathbf{d}', \mathbf{x}', \mathbf{g}'/\mathbf{d}, \mathbf{x}, \mathbf{g}], \sigma \cup \sigma_{\mathrm{d}'\mathrm{x}'}\rangle \overset{\rho(0)}{\rightsquigarrow}$. By Rule 46, we obtain $(C, J, L, H, R) \Vdash \langle [\![_{\mathrm{V}}\, \sigma_{\mathrm{dx}_\perp}, \{\mathbf{x}\}, \{\mathbf{g}\} \mid q\,]\!], \sigma\rangle \overset{\rho\downarrow(\mathrm{dom}(\sigma)\cup\dot{C}\cup L)(0)}{\rightsquigarrow}$.
- Rules 49 and 52. The proofs are similar. We only give the proof for Rule 49. Then $p \equiv [\![_{\mathrm{H}}\, \{\mathbf{h}\} \mid q\,]\!]$ for some $\mathbf{h}$, q, $E = (C, J, L, H, R)$, $(C,\ J,\ L, H \cup \{\mathbf{h}'\}, R) \Vdash \langle q[\mathbf{h}'/\mathbf{h}], \sigma\rangle \overset{t,\rho}{\longmapsto} \langle q', \sigma'\rangle$ for some q'. Note that the syntactical equality of p' is not given, because it is irrelevant for this proof. By induction we then have $\langle q[\mathbf{h}'/\mathbf{h}], \sigma,\ (C,\ J,\ L, H \cup \{\mathbf{h}'\}, R)\ \rangle \overset{\rho(0)}{\rightsquigarrow}$. By Rule 50, we obtain $(C, J, L, H, R) \Vdash \langle [\![_{\mathrm{H}}\, \{\mathbf{h}\} \mid q\,]\!], \sigma\rangle \overset{\rho(0)}{\rightsquigarrow}$.

The rules that have not been considered could not have been applied last since they conclude an action transition or a consistency predicate.

A.4 The proof of Lemma 4

Let p be a closed process term, σ be a valuation, C, J, W, L be sets of various classes of χ variables such that J and $W \subseteq \mathrm{dom}(\sigma) \setminus \{\mathsf{time}\}$, H be a set of channels, R be a recursion definition, and ξ be an extended valuation. Then

$$\langle p, \sigma, (C, J, L, H, R)\rangle \overset{\xi}{\rightsquigarrow} \Leftrightarrow \langle p, \sigma, (C, J \cup W, L, H, R)\rangle \overset{\xi}{\rightsquigarrow}.$$

PROOF. The proof is trivial. The domain of the extended valuation ξ is given by $\mathrm{dom}(\sigma) \cup \dot{C} \cup L$ for all χ consistency predicate rules. Hence, any variation in the set of jumping variables in the environment of a consistent χ process is irrelevant for the consistency predicate.

B Proofs of properties of the Chi operators

In this section, the outline of the proofs for the properties in Section 5.4 is given. For all of these properties, the proofs follow the same lines. A relation R is defined containing at least all closed instantiations of the property to be proved. Then, it must be shown that this relation is a stateless bisimulation. For this in principle for each pair of closed process terms $(p, q) \in R$, it has to be shown that it satisfies the six conditions of Definition 5. Often, the relation R contains pairs of the form (i_d, i_d). Since the proofs are trivial for such pairs these are omitted. As the deduction rules of χ are such that the environment does not change in a transition, we only consider those cases in the proofs. As a consequence we use the notation $E \Vdash$ as much as possible.

Since a deduction rule A may consist of some sub-deduction rules, we use the notation Rule A.i.s to indicate the sub-deduction rule that has been applied in the proofs, where A represents a deduction rule number, i represents an index, and s indicates the left or right result.

Consider the following deduction rule A:

$$\frac{\begin{array}{c}\langle p,\sigma,E\rangle \xrightarrow{\xi,a,\xi'} \langle \begin{array}{c} p'_{11} \\ \vdots \\ p'_{1n} \end{array}, \sigma', E\rangle, \langle q,\sigma,E\rangle \xrightarrow{\xi,a,\xi'} \langle \begin{array}{c} q'_{11} \\ \vdots \\ q'_{1n} \end{array}, \sigma', E\rangle \end{array}}{\begin{array}{c}\langle l,\sigma,E\rangle \xrightarrow{\xi,a,\xi'} \langle \begin{array}{c} l'_{11} \\ \vdots \\ l'_{1n} \end{array}, \sigma', E\rangle, \langle r,\sigma,E\rangle \xrightarrow{\xi,a,\xi'} \langle \begin{array}{c} r'_{11} \\ \vdots \\ r'_{1n} \end{array}, \sigma', E\rangle \end{array}} \text{(A)}$$

Rule A.1.l refers to the following sub-deduction rule of deduction rule A:

$$\frac{\langle p,\sigma,E\rangle \xrightarrow{\xi,a,\xi'} \langle p'_{11},\sigma',E\rangle, \langle q,\sigma,E\rangle \xrightarrow{\xi,a,\xi'} \langle q'_{11},\sigma',E\rangle}{\langle l,\sigma,E\rangle \xrightarrow{\xi,a,\xi'} \langle l'_{11},\sigma',E\rangle}$$

Similarly, Rule A.n.r refers to the following sub-deduction rule of deduction rule A:

$$\frac{\langle p,\sigma,E\rangle \xrightarrow{\xi,a,\xi'} \langle p'_{1n},\sigma',E\rangle, \langle q,\sigma,E\rangle \xrightarrow{\xi,a,\xi'} \langle q'_{1n},\sigma',E\rangle}{\langle r,\sigma,E\rangle \xrightarrow{\xi,a,\xi'} \langle r'_{1n},\sigma',E\rangle}$$

Note that i and/or s can be omitted in the notation Rule A.i.s when there is no such a sub-deduction rule.

Lemma 14 *For arbitrary closed process term p we have*

$$\text{true} \curvearrowright p \leftrightarrows p.$$

PROOF. Let $R = \{(\text{true} \curvearrowright p, p) \mid p \in P\} \cup \{(i_d, i_d) \mid i_d \in P\}$. The proofs of conditions 1 – 3 are similar to the proofs of conditions 1 – 3 of Lemma 21 (except the premise $\xi \models b$ is replaced by $\xi \models u$). The proofs of conditions 4 and 5 are similar to the proofs of conditions 2 and 3 (notice that the premise $\xi \models u$ is replaced by $\rho(0) \models u$ in the proofs). The proofs of condition 6 are similar to the proofs of condition 6 of Lemma 21 (except Rule 24 has not been applied, and the premise $\xi \models b$ is replaced by $\xi \models u$ in the proofs).

Lemma 15 *For arbitrary closed process term p we have*

$$\text{false} \curvearrowright p \leftrightarrows \bot.$$

PROOF. The fact that there are no action transition rules, time transition rules and consistency predicate rules defined for $\curvearrowright$ in which the initialization predicate is not satisfied, also indicates that false $\curvearrowright p$ cannot perform any transition. Therefore, the conditions 1 – 6 hold trivially.

Lemma 16 *For arbitrary predicate u we have*

$$u \curvearrowright u \leftrightarrows u.$$

PROOF. Let $R = \{(u \curvearrowright u, u) \mid \text{predicate } u\} \cup \{(i_d, i_d) \mid i_d \in P\}$. The fact that there are no action transition rules defined for u, also indicates that $u \curvearrowright u$ has no action transitions. Therefore, the conditions 1 – 3 hold trivially.

Condition 4: We assume $(C, J, L, H, R) \Vdash \langle u \curvearrowright u, \sigma\rangle \xrightarrow{t,\rho} \langle k_1, \sigma'\rangle$ for some C, J, L, H, R, σ, t, ρ, k_1, σ', which means that Rule 14 has been applied necessarily. Then, $(C, J, L, H, R) \Vdash \langle u, \sigma\rangle \xrightarrow{t,\rho} \langle k_1, \sigma'\rangle$ and $\rho(0) \models u$. For $(C, J, L, H, R) \Vdash \langle u, \sigma\rangle \xrightarrow{t,\rho} \langle k_1, \sigma'\rangle$, Rule 3 has been applied necessarily. Then, $\rho \in \Omega_{FG}(\sigma, C, L, u, t)$, $\sigma' = \rho_\sigma(t)$ and $k_1 \equiv u$. Using Rule 3, we obtain $(C, J, L, H, R) \Vdash \langle u, \sigma\rangle \xrightarrow{t,\rho} \langle u, \rho_\sigma(t)\rangle$ and observe that $(u, u) \in R$.

Condition 5: We assume $(C, J, L, H, R) \Vdash \langle u, \sigma\rangle \xrightarrow{t,\rho} \langle k_1, \sigma'\rangle$ for some C, J, L, H, R, σ, t, ρ, k_1, σ', which means that Rule 3 has been applied necessarily. Then, $\rho \in \Omega_{FG}(\sigma, C, L, u, t)$, $\sigma' = \rho_\sigma(t)$ and $k_1 \equiv u$. We know that $\forall_{s\in[0,t]}\, \rho(s) \models u$ (from the definition of the function Ω_{FG}). Hence, we also have $\rho(0) \models u$. Using

Rule 14, we obtain $(C, J, L, H, R) \Vdash \langle u \curvearrowright u, \sigma\rangle \stackrel{t,\rho}{\longmapsto} \langle u, \rho_\sigma(t)\rangle$ and observe that $(u, u) \in R$.

Condition 6: First, we assume $(C, J, L, H, R) \Vdash \langle u \curvearrowright u, \sigma\rangle \stackrel{\xi}{\rightsquigarrow}$ for some C, J, L, H, R, σ, ξ, which means that Rule 15 has been applied necessarily. Then, $(C, J, L, H, R) \Vdash \langle u, \sigma\rangle \stackrel{\xi}{\rightsquigarrow}$ and $\xi \models u$. Second, we assume $(C, J, L, H, R) \Vdash \langle u, \sigma\rangle \stackrel{\xi}{\rightsquigarrow}$ for some C, J, L, H, R, σ, ξ, which means Rule 4 has been applied necessarily. Then, $\xi = \sigma \cup \xi^{\dot{C}L}$ for some $\xi^{\dot{C}L}$ and $\sigma \cup \xi^{\dot{C}L} \models u$. According to Rule 15, we get $(C, J, L, H, R) \Vdash \langle u \curvearrowright u, \sigma\rangle \stackrel{\sigma\cup\xi^{\dot{C}L}}{\rightsquigarrow}$.

Lemma 17 *For arbitrary closed process term p and arbitrary predicates u, u' we have*

$$u \curvearrowright (u' \curvearrowright p) \underline{\leftrightarrow} (u \wedge u') \curvearrowright p.$$

PROOF. Let $R = \{(u \curvearrowright (u' \curvearrowright p), (u \wedge u') \curvearrowright p) \mid p \in P, \text{predicates } u, u'\} \cup \{(i_d, i_d) \mid i_d \in P\}$.

Condition 1: First, we assume $E \Vdash \langle u \curvearrowright (u' \curvearrowright p), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ for some E, σ, ξ, a, ξ', σ', which means Rule 13.1 has been applied necessarily. Then, $E \Vdash \langle u' \curvearrowright p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ and $\xi \models u$. Again, Rule 13.1 has been applied necessarily. Therefore, we have $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ and $\xi \models u'$. From $\xi \models u$ and $\xi \models u'$ we get $\xi \models u \wedge u'$. Using Rule 13.1, we obtain $E \Vdash \langle (u \wedge u') \curvearrowright p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$. Second, we assume $E \Vdash \langle (u \wedge u') \curvearrowright p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ for some E, σ, ξ, a, ξ', σ', which means Rule 13.1 has been applied necessarily. Thus, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ and $\xi \models u \wedge u'$. From $\xi \models u \wedge u'$ we obtain $\xi \models u$ and $\xi \models u'$. Using Rule 13.1, we obtain $E \Vdash \langle u' \curvearrowright p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$. Again using Rule 13.1, we obtain $E \Vdash \langle u \curvearrowright (u' \curvearrowright p), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$.

Condition 2: We assume $E \Vdash \langle u \curvearrowright (u' \curvearrowright p), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ for some E, σ, ξ, a, ξ', k_1, σ', which means Rule 13.2 has been applied necessarily. Thus, $E \Vdash \langle u' \curvearrowright p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and $\xi \models u$. Again, Rule 13.2 has been applied necessarily. Therefore, we have $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and $\xi \models u'$. From $\xi \models u$ and $\xi \models u'$, we obtain $\xi \models u \wedge u'$. Using Rule 13.2, we get $E \Vdash \langle (u \wedge u') \curvearrowright p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and observe that $(k_1, k_1) \in R$.

Condition 3: We assume $E \Vdash \langle (u \wedge u') \curvearrowright p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ for some E, σ, ξ, a, ξ', k_1, σ', which means Rule 13.2 has been applied necessarily. Therefore, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and $\xi \models u \wedge u'$. From $\xi \models u \wedge u'$, we also have $\xi \models u$ and $\xi \models u'$. Using Rule 13.2 we obtain $E \Vdash \langle u' \curvearrowright p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$. Again

using Rule 13.2 we obtain $E \Vdash \langle u \curvearrowright (u' \curvearrowright p), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and observe that $(k_1, k_1) \in R$.

Condition 4: We assume $E \Vdash \langle u \curvearrowright (u' \curvearrowright p), \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma'\rangle$ for some E, σ, t, ρ, k_1, σ', which means that Rule 14 has been applied necessarily. Then, $E \Vdash \langle u' \curvearrowright p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma'\rangle$ and $\rho(0) \models u$. For $E \Vdash \langle u' \curvearrowright p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma'\rangle$, Rule 14 has been applied necessarily. Then, $E \Vdash \langle p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma'\rangle$ and $\rho(0) \models u'$. From $\rho(0) \models u$ and $\rho(0) \models u'$, we obtain $\rho(0) \models u \wedge u'$. Using Rule 14, we get $E \Vdash \langle (u \wedge u') \curvearrowright p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle (u \wedge u') \curvearrowright k_1, \sigma'\rangle$ and observe that $(k_1, k_1) \in R$.

Condition 5: We assume $E \Vdash \langle (u \wedge u') \curvearrowright p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma'\rangle$ for some E, σ, t, ρ, k_1, σ', which means that Rule 14 has been applied necessarily. Then, $E \Vdash \langle p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma'\rangle$ and $\rho(0) \models u \wedge u'$. From $\rho(0) \models u \wedge u'$, we can also have $\rho(0) \models u$ and $\rho(0) \models u'$. Using Rule 14, we obtain $E \Vdash \langle u' \curvearrowright p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma'\rangle$. Again, using Rule 14 we get $E \Vdash \langle u \curvearrowright (u' \curvearrowright p), \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma'\rangle$ and observe that $(k_1, k_1) \in R$.

Condition 6: First, we assume $E \Vdash \langle u \curvearrowright (u' \curvearrowright p), \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some E, σ, ξ, which means that Rule 15 has been applied necessarily. Then, $E \Vdash \langle u' \curvearrowright p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and $\xi \models u$. For $E \Vdash \langle u' \curvearrowright p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, Rule 15 has been applied necessarily. Then $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and $\xi \models u'$. From $\xi \models u$ and $\xi \models u'$, we can have $\xi \models u \wedge u'$. Using Rule 15, we obtain $E \Vdash \langle (u \wedge u') \curvearrowright p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Second, we assume $E \Vdash \langle (u \wedge u') \curvearrowright p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some E, σ, ξ, which means Rule 15 has been applied necessarily. Then, $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and $\xi \models (u \wedge u')$. From $\xi \models (u \wedge u')$, we get $\xi \models u$ and $\xi \models u'$. According to Rule 15, we obtain $E \Vdash \langle u' \curvearrowright p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 15, we get $E \Vdash \langle u \curvearrowright (u' \curvearrowright p), \sigma\rangle \overset{\xi}{\rightsquigarrow}$.

B.2 Properties of alternative composition

Lemma 18 (Idempotency of alternative composition) *For closed term p we have*

$$p \mathbin{[\!]} p \leftrightarroweq p.$$

PROOF. Let $R = \{(p \mathbin{[\!]} p, p) \mid p \in P\} \cup \{(i_d, i_d) \mid i_d \in P\}$.

Condition 1: First, we assume $E \Vdash \langle p \mathbin{[\!]} p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$ for some E, σ, ξ, a, ξ', σ', which means that Rule 25.1.l or Rule 25.1.r has been applied necessarily. Since the left and right argument of the $[\!]$ are the same, we only give the proofs in which Rule 25.1.l has been applied. Then, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$.

Second, we assume $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ for some E, σ, ξ, a, ξ', σ'. We know that $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ (see also Lemma 2). Using Rule 25.1.l, we obtain $E \Vdash \langle p \between p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$.

Condition 2: We assume $E \Vdash \langle p \between p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ for some E, σ, ξ, a, ξ', k_1, σ', which means that Rule 25.2.l or Rule 25.2.r has been applied necessarily. Since the left and right argument of the $\between$ are the same, we only the proofs in which Rule 25.1.l has been applied. Then, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and observe that $(k_1, k_1) \in R$.

Condition 3: We assume $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ for some E, σ, ξ, a, ξ', k_1, σ'. We also know that $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ (see also Lemma 2). Using Rule 25.2.l, we obtain $E \Vdash \langle p \between p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and observe that $(k_1, k_1) \in R$.

Condition 4: We assume $E \Vdash \langle p \between p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma'\rangle$ for some E, σ, t, ρ, k_1, σ', which means Rule 26 has been applied necessarily. Then, we get $E \Vdash \langle p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_p, \sigma'\rangle$ for some k_p such that $k_1 \equiv k_p \between k_p$. Take $k_2 \equiv k_p$ and observe that $(k_1, k_2) \in R$.

Condition 5: We assume $E \Vdash \langle p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma'\rangle$ for some E, σ, t, ρ, k_1, σ'. Using Rule 26, we obtain $E \Vdash \langle p \between p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1 \between k_1, \sigma'\rangle$. Take $k_2 \equiv k_1 \between k_1$ and observe that $(k_2, k_1) \in R$.

Condition 6: First, we assume $E \Vdash \langle p \between p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some E, σ, ξ, which means Rule 27 has been applied necessarily. Then, we get $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Second, we assume $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some E, σ, ξ. Using Rule 27, we obtain $E \Vdash \langle p \between p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$.

Lemma 19 (Commutativity of alternative composition) *For arbitrary closed process terms p and q we have*

$$p \between q \leftrightarrows q \between p.$$

PROOF. Let $R = \{(p \between q, q \between p) \mid p, q \in P\} \cup \{(i_d, i_d) \mid i_d \in P\}$. Since the deduction rules for $\between$ are symmetrical w.r.t. the left and right argument, obviously all conditions are met.

Lemma 20 (Associativity of alternative composition) *For closed process terms p, q and r we have*

$$(p \between q) \between r \leftrightarrows p \between (q \between r).$$

PROOF. Let $R = \{((p \parallel q) \parallel r, p \parallel (q \parallel r)) \mid p, q, r \in P\} \cup \{(i_d, i_d) \mid i_d \in P\}$. The proof of the left implication of condition 1 is similar to the proof of the right implication. The proofs of conditions 3 and 5 are similar to the proofs of conditions 2 and 4. The proof of the left implication of condition 6 is similar to the proof of the right implication.

Condition 1: We assume $E \Vdash \langle (p \parallel q) \parallel r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$ for some E, σ, ξ, a, ξ', σ', which means that Rule 25.1.l or Rule 25.1.r has been applied necessarily. Hence, we distinguish two cases:

(1) Rule 25.1.l has been applied. Then, $E \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$ and $E \Vdash \langle r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. For $E \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$, this means that again either Rule 25.1.l or Rule 25.1.r has been applied necessarily. Hence, we can further distinguish two cases:
   (a) Rule 25.1.l has been applied. Then, $E \Vdash \langle p, \sigma\rangle \overset{\xi,a,\xi'}{\rightsquigarrow} \langle \checkmark, \sigma'\rangle$ and $E \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 27, we obtain $E \Vdash \langle q \parallel r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. We further get $E \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$ using Rule 25.1.l.
   (b) Rule 25.1.r has been applied. Then, $E \Vdash \langle q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$ and $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 25.1.l, we obtain $E \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$. Applying Rule 25.1.r, we obtain $E \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$.
(2) Rule 25.1.r has been applied. The proof is similar to the previous case.

Condition 2: We assume $E \Vdash \langle (p \parallel q) \parallel r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ for some E, σ, ξ, a, ξ', k_1, σ', which means that either Rule 25.2.l or Rule 25.2.r has been applied necessarily. Hence, we distinguish two cases:

(1) Rule 25.2.l has been applied. Then $E \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and $E \Vdash \langle r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. For $E \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$, this means that again either Rule 25.2.l or Rule 25.2.r has been applied necessarily. Hence, we again distinguish two cases:
   (a) Rule 25.2.l has been applied. Then, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and $E \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 27, we obtain $E \Vdash \langle q \parallel r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. We further get $E \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ using Rule 25.2.l and observe that $(k_1, k_1) \in R$.
   (b) Rule 25.2.r has been applied. Then $E \Vdash \langle q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. We get $E \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ using Rule 25.2.l. Applying Rule 25.2.r, we obtain $E \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and observe that $(k_1, k_1) \in R$.
(2) Rule 25.2.r has been applied. The proof is similar to the previous case.

Condition 4: We assume $E \Vdash \langle (p \parallel q) \parallel r, \sigma\rangle \xmapsto{t,\rho} \langle k_1, \sigma'\rangle$ for some E, σ, t,

ρ, k_1, σ', which means Rule 26 has been applied necessarily. Then $E \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{t,\rho} \langle k_{pq}, \sigma'\rangle$ and $E \Vdash \langle r, \sigma\rangle \xrightarrow{t,\rho} \langle k_r, \sigma'\rangle$ for some k_{pq} and k_r such that $k_1 \equiv k_{pq} \parallel k_r$. For $E \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{t,\rho} \langle k_{pq}, \sigma'\rangle$, we obtain $E \Vdash \langle p, \sigma\rangle \xrightarrow{t,\rho} \langle k_p, \sigma'\rangle$ and $E \Vdash \langle q, \sigma\rangle \xrightarrow{t,\rho} \langle k_q, \sigma'\rangle$ for some k_p, k_q such that $k_{pq} \equiv k_p \parallel k_q$ (using Rule 26). Applying Rule 26, we get $E \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{t,\rho} \langle k_q \parallel k_r, \sigma'\rangle$. Again, due to Rule 26, we can have $E \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{t,\rho} \langle k_p \parallel (k_q \parallel k_r), \sigma'\rangle$. Note that $k_1 \equiv (k_p \parallel k_q) \parallel k_r$. Take $k_2 \equiv k_p \parallel (k_q \parallel k_r)$ and observe that $(k_1, k_2) \in R$.

Condition 6: We assume $E \Vdash \langle (p \parallel q) \parallel r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, which means Rule 27 has been applied necessarily. Then $E \Vdash \langle p \parallel q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and $E \Vdash \langle r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. For $E \Vdash \langle p \parallel q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, we obtain $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and $E \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ (see also Rule 27). Applying Rule 27, we get $E \Vdash \langle q \parallel r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Again, due to Rule 27, we can have $E \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \overset{\xi}{\rightsquigarrow}$.

B.3 Properties of guard operator

Lemma 21 *For arbitrary closed process term p we have*

$$\text{true} \rightarrow p \leftrightarrows p.$$

PROOF. Let $R = \{(\text{true} \rightarrow p, p) \mid p \in P\} \cup \{(i_d, i_d) \mid i_d \in P\}$.

Condition 1: First, we assume $E \Vdash \langle \text{true} \rightarrow p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ for some $E, \sigma, \xi, a, \xi', \sigma'$, which means that Rule 20.1 has been applied necessarily. Then, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$. Second, we assume $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ for some $E, \sigma, \xi, a, \xi', \sigma'$. We also know that $\xi \models \text{true}$, and obtain $E \Vdash \langle \text{true} \rightarrow p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ using Rule 20.1.

Condition 2: We assume $E \Vdash \langle \text{true} \rightarrow p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ for some E, σ, ξ, a, ξ', k_1, σ', which means that Rule 20.2 has been applied necessarily. Then, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and observe that $(k_1, k_1) \in R$.

Condition 3: We assume $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ for some E, σ, ξ, a, ξ', k_1, σ'. We also know that $\xi \models \text{true}$. We obtain $E \Vdash \langle \text{true} \rightarrow p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ using Rule 20.2 and observe that $(k_1, k_1) \in R$.

Condition 4: We assume $E \Vdash \langle \text{true} \rightarrow p, \sigma\rangle \xrightarrow{t,\rho} \langle k_1, \sigma'\rangle$ for some E, σ, t, ρ, k_1, σ', which means Rule 21 has been applied necessarily. Notice that Rule 22 cannot be applied, because the premise $\forall_{s\in(0,t)}\, \rho(s) \models \neg\text{true}$ does

not hold. Then $E \Vdash \langle p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_p, \sigma'\rangle$ for some k_p such that $k_1 \equiv \text{true} \rightarrow k_p$ and $\forall_{s\in[0,t]}\, \rho(s) \models \text{true}$. Take $k_2 \equiv k_p$ and observe that $(k_1, k_2) \in R$.

Condition 5: We assume $E \Vdash \langle p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma'\rangle$ for some E, σ, t, ρ, k_1, σ'. We also know that $\forall_{s\in[0,t]}\, \rho(s) \models \text{true}$. We obtain $E \Vdash \langle \text{true} \rightarrow p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle \text{true} \rightarrow k_1, \sigma'\rangle$ using Rule 21. Take $k_2 \equiv \text{true} \rightarrow k_1$ and observe that $(k_2, k_1) \in R$.

Condition 6: First, we assume $E \Vdash \langle \text{true} \rightarrow p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some E, σ, ξ, which means Rule 23 has been applied necessarily. Notice that Rule 24 cannot have applied, because the premise $\sigma \cup \xi^{\dot{C}L} \models \neg\text{true}$ does not hold. Then $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Second, we assume $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some E, σ, ξ. We also know $\xi \models \text{true}$. We obtain $E \Vdash \langle \text{true} \rightarrow p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ using Rule 23.

Lemma 22 *For arbitrary closed process term p we have*

$$\text{false} \rightarrow p \leftrightarrows \text{true}.$$

PROOF. Let $R = \{(\text{false} \rightarrow p, \text{true}) \mid p \in P\} \cup \{(i_d, i_d) \mid i_d \in P\}$. Since there no are action transition rules defined for guard that evaluates to false in the extended valuation (i.e. $\xi \models \text{false}$), and for the process term true also no action transition rules are defined, the conditions 1 – 3 hold trivially.

Condition 4: We assume $(C, J, L, H, R) \Vdash \langle \text{false} \rightarrow p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma'\rangle$ for some C, J, L, H, R, σ, t, ρ, k_1, σ' which means Rule 22 has been applied necessarily. Notice that Rule 21 cannot be applied, because the premise $\forall_{s\in[0,t]}\, \rho(s) \models \text{false}$ does not hold. Then $k_1 \equiv \text{false} \rightarrow p$, $\sigma' = \rho_\sigma(t)$, $\rho \in \Omega_{\sigma E t}$ and $\forall_{s\in(0,t)}\, \rho(s) \models \neg\text{false}$. For $\rho \in \Omega_{\sigma E t}$, we can have $(C, J, L, H, R) \Vdash \langle \text{true}, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle \text{true}, \rho_\sigma(t)\rangle$ (see also Rule 3). Take $k_2 \equiv \text{true}$ and observe that $(k_1, k_2) \in R$.

Condition 5: We assume $(C, J, L, H, R) \Vdash \langle \text{true}, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma'\rangle$ for some C, J, L, H, R, σ, t, ρ, k_1, σ', which means Rule 3 has been applied necessarily. Then $k_1 \equiv \text{true}$, $\sigma' = \rho_\sigma(t)$, and $\rho \in \Omega_{FG}(\sigma, C, L, \text{true}, t)$. We know that $\forall_{s\in(0,t)}\, \rho(s) \models \neg\text{false}$, $\rho(0) \models \text{false} \Rightarrow (C, J, L, H, R) \Vdash \langle p, \sigma\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle p', \sigma''\rangle$ for some p', σ'', and $\rho(t) \models \text{false} \Rightarrow (C, J, L, H, R) \Vdash \langle p, \rho_\sigma(t)\rangle \overset{\rho(t)}{\rightsquigarrow}$ (since the left-hand sides of the implications are false, these two implications hold trivially). Using Rule 22, we obtain $(C, J, L, H, R) \Vdash \langle \text{false} \rightarrow p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle \text{false} \rightarrow p, \sigma'\rangle$. Take $k_2 \equiv \text{false} \rightarrow p$ and observe that $(k_2, k_1) \in R$.

Condition 6: First, we assume $(C, J, L, H, R) \Vdash \langle \text{false} \rightarrow p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some C, J, L, H, R, σ, ξ, which means Rule 24 has been applied necessarily. Notice that Rule 23 cannot have been applied, because $\xi \models \text{false}$ does not hold. Then, $\xi = \sigma \cup \xi^{\dot{C}L}$ for some $\sigma \cup \xi^{\dot{C}L}$. We know that $\sigma \cup \xi^{\dot{C}L} \models \text{true}$. Using Rule 4, we

obtain $(C, J, L, H, R) \Vdash \langle \text{true}, \sigma \rangle \overset{\sigma \cup \xi^{\dot{C}L}}{\rightsquigarrow}$. Second, we assume $(C, J, L, H, R) \Vdash \langle \text{true}, \sigma \rangle \overset{\xi}{\rightsquigarrow}$ for some C, J, L, H, R, σ, ξ, which means Rule 4 has been applied necessarily. Then, $\xi = \sigma \cup \xi^{\dot{C}L}$ for some $\sigma \cup \xi^{\dot{C}L}$ and $\sigma \cup \xi^{\dot{C}L} \models \text{true}$. We also know that $\sigma \cup \xi^{\dot{C}L} \models \neg\text{false}$. Using Rule 24 we get $(C, J, L, H, R) \Vdash \langle \text{false} \to p, \sigma \rangle \overset{\sigma \cup \xi^{\dot{C}L}}{\rightsquigarrow}$.

Lemma 23 *For arbitrary closed process term p and arbitrary guard b we have*

$$b \to \bot \underline{\leftrightarrow} \neg b.$$

PROOF. Let $R = \{(b \to \bot, \neg b) \mid p \in P, \text{guard } b\} \cup \{(i_d, i_d) \mid i_d \in P\}$. Since there are no action transition rules defined for $\bot$, also $b \to \bot$ has no action transition rules defined, and there are no action transition rules defined for delay predicates, the conditions 1 – 3 hold trivially.

Condition 4: We assume $(C, J, L, H, R) \Vdash \langle b \to \bot, \sigma \rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma' \rangle$ for some C, J, L, H, R, σ, t, ρ, k_1, σ', which means that either Rule 21 or Rule 22 has been applied necessarily. Then we can distinguish two cases:

(1) Rule 21 has been applied. Then, $(C, J, L, H, R) \Vdash \langle \bot, \sigma \rangle \overset{t,\rho}{\longmapsto} \langle k_p, \sigma' \rangle$ for some k_p. This leads a contradiction, because $\bot$ cannot perform any action transitions. Thus, Rule 21 cannot have been applied.

(2) Rule 22 has been applied. Then, $k_1 \equiv b \to \bot$ and $\sigma' = \rho_\sigma(t)$, $\rho \in \Omega_{\sigma E t}$, $\forall_{s \in (0,t)}\ \rho(s) \models \neg b, \rho(0) \models b \Rightarrow (C, J, L, H, R) \Vdash \langle \bot, \sigma \rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle z, \sigma'' \rangle$ for some z, σ'' and $\rho(t) \models b \Rightarrow (C, J, L, H, R) \Vdash \langle \bot, \rho_\sigma(t) \rangle \overset{\rho(t)}{\rightsquigarrow}$. From the facts $\rho(0) \models b \Rightarrow (C, J, L, H, R) \Vdash \langle \bot, \sigma \rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle z, \sigma'' \rangle$ and $\rho(t) \models b \Rightarrow (C, J, L, H, R) \Vdash \langle \bot, \rho_\sigma(t) \rangle \overset{\rho(t)}{\rightsquigarrow}$, we get $\rho(0) \models \neg b$ and $\rho(t) \models \neg b$, since the right-hand side of these implications are false (due to $\bot$ cannot perform any transition). Thus, we have $\forall_{s \in [0,t]}\, \rho(s) \models \neg b$. Hence, we can also obtain the following transition $(C, J, L, H, R) \Vdash \langle \neg b, \sigma \rangle \overset{t,\rho}{\longmapsto} \langle \neg b, \rho_\sigma(t) \rangle$ (see also Rule 3). Take $k_2 \equiv \neg b$ and observe that $(k_1, k_2) \in R$.

Condition 5: We assume $(C, J, L, H, R) \Vdash \langle \neg b, \sigma \rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma' \rangle$ for some C, J, L, H, R, σ, t, ρ, k_1, σ', which means that Rule 3 has been applied necessarily. Then, $k_1 \equiv \neg b$, $\rho \in \Omega_{FG}(\sigma, C, L, \neg b, t)$ and $\sigma' = \rho_\sigma(t)$. From $\rho \in \Omega_{FG}(\sigma, C, L, \neg b, t)$, we know that $\forall_{s \in [0,t]}\, \rho(s) \models \neg b$ and $\bot$ also cannot perform any transition. Then the following premises $\rho(0) \models b \Rightarrow (C, J, L, H, R) \Vdash \langle \bot, \sigma \rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle z, \sigma'' \rangle$ for some z, σ'', and $\rho(t) \models b \Rightarrow (C, J, L, H, R) \Vdash \langle \bot, \rho_\sigma(t) \rangle \overset{\rho(t)}{\rightsquigarrow}$ hold (because the left-hand side of the implications are false). Using Rule 22, we obtain $(C, J, L, H, R) \Vdash \langle b \to \bot, \sigma \rangle \overset{t,\rho}{\longmapsto} \langle b \to \bot, \rho_\sigma(t) \rangle$. Take $k_2 \equiv b \to \bot$ and observe that $(k_2, k_1) \in R$.

Condition 6: First, we assume $(C, J, L, H, R) \Vdash \langle b \to \bot, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some C, J, L, H, R, σ, ξ, which means that Rule 24 has been applied necessarily. Notice that Rule 23 cannot be applied, because the premise $(C, J, L, H, R) \Vdash \langle \bot, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ does not hold. Then $\xi = \sigma \cup \xi^{\dot{C}L}$ for some $\sigma \cup \xi^{\dot{C}L}$ and $\sigma \cup \xi^{\dot{C}L} \models \neg b$. Applying Rule 4, we get $(C, J, L, H, R) \Vdash \langle \neg b, \sigma\rangle \overset{\sigma\cup\xi^{\dot{C}L}}{\rightsquigarrow}$. Second, we assume $(C, J, L, H, R) \Vdash \langle \neg b, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some C, J, L, H, R, σ, ξ, which means Rule 4 has been applied necessarily. Then $\xi = \sigma \cup \xi^{\dot{C}L}$ for some $\sigma \cup \xi^{\dot{C}L}$ and $\sigma \cup \xi^{\dot{C}L} \models \neg b$. Using Rule 24, we obtain $(C, J, L, H, R) \Vdash \langle b \to \bot, \sigma\rangle \overset{\sigma\cup\xi^{\dot{C}L}}{\rightsquigarrow}$.

Lemma 24 (Distributivity of guard over alternative comp.) *For arbitrary closed process terms p and q and arbitrary guard b we have*

$$b \to (p \,[\!]\, q) \leftrightarrows b \to p \,[\!]\, b \to q.$$

PROOF. Let $R = \{(b \to (p \,[\!]\, q), b \to p \,[\!]\, b \to q) \mid p, q \in P, \text{guard } b\} \cup \{(i_d, i_d) \mid i_d \in P\}$.

Condition 1: First, we assume $E \Vdash \langle b \to (p \,[\!]\, q), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ for some E, σ, ξ, a, ξ', σ', which means that Rule 20.1 has been applied necessarily. Then, $E \Vdash \langle p \,[\!]\, q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ and $\xi \models b$. For $E \Vdash \langle p \,[\!]\, q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$, we distinguish two cases:

(1) Rule 25.1.l has been applied. Then $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ and $E \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 20.1, we have $E \Vdash \langle b \to p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$. We also obtain $E \Vdash \langle b \to q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ using Rule 23. Applying Rule 25.1.l, we get $E \Vdash \langle b \to p \,[\!]\, b \to q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$.
(2) Rule 25.1.r has been applied. The proof is similar to the proof of the previous case.

Second, we assume $(C, J, L, H, R) \Vdash \langle b \to p \,[\!]\, b \to q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ for some C, J, L, H, R, σ, ξ, a, ξ', σ', which means that Rule 25.1.l or Rule 25.1.r has been applied necessarily. We distinguish two cases:

(1) Rule 25.1.l has been applied. Then $(C, J, L, H, R) \Vdash \langle b \to p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ and $(C, J, L, H, R) \Vdash \langle b \to q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. According to Rule 20.1, we must have $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ and $\xi \models b$. For $E = C$, J, L, H, R, $\langle b \to q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, which means that either Rule 23 or Rule 24 has been applied necessarily. We distinguish two cases:
   (a) Rule 23 has been applied. Then $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Applying Rule 25.1.l, we can have $(C, J, L, H, R) \Vdash \langle p \,[\!]\, q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$. Due

to Rule 20.1, we finally get $(C, J, L, H, R) \Vdash \langle b \to (p \,[\!]\, q), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$.

(b) Rule 24 has been applied. Then $\xi = \sigma \cup \xi^{\dot{C}L}$ for some $\sigma \cup \xi^{\dot{C}L}$ and $\sigma \cup \xi^{\dot{C}L} \models \neg b$. This leads to a contradiction. Therefore this case cannot occur.

(2) Rule 25.1.r has been applied. The proof is similar to the proof of the previous case.

Condition 2: We assume $E \Vdash \langle b \to (p \,[\!]\, q), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ for some E, σ, ξ, a, ξ', k_1, σ', which means that Rule 20.2 has been applied necessarily. Then, $E \Vdash \langle p \,[\!]\, q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and $\xi \models b$. For $E \Vdash \langle p \,[\!]\, q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$, we distinguish two cases:

(1) Rule 25.2.l has been applied. Then $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and $E \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 20.1, we have $E \Vdash \langle b \to p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$. We also obtain $E \Vdash \langle b \to q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ using Rule 23. Applying Rule 25.2.l, we get $E \Vdash \langle b \to p \,[\!]\, b \to q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and observe that $(k_1, k_1) \in R$.

(2) Rule 25.1.r has been applied. The proof is similar to the proof of the previous case.

Condition 3: We assume $(C, J, L, H, R) \Vdash \langle b \to p \,[\!]\, b \to q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ for some E, σ, ξ, a, ξ', k_1, σ', which means that Rule 25.1.l or Rule 25.1.r has been applied necessarily. We distinguish two cases:

(1) Rule 25.2.l has been applied. Then $(C, J, L, H, R) \Vdash \langle b \to p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and $(C, J, L, H, R) \Vdash \langle b \to q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. According to Rule 20.2, we must have $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and $\xi \models b$. For $(C, J, L, H, R) \Vdash \langle b \to q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, which means that either Rule 23 or Rule 24 has been applied necessarily. We distinguish two cases:

(a) Rule 23 has been applied. We have $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ using Rule 23. Applying Rule 25.2.l, we have $(C, J, L, H, R) \Vdash \langle p \,[\!]\, q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$. Due to Rule 20.2, we finally get $(C, J, L, H, R) \Vdash \langle b \to (p \,[\!]\, q), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and observe that $(k_1, k_1) \in R$.

(b) Rule 24 has been applied. Then $\xi = \sigma \cup \xi^{\dot{C}L}$ for some $\sigma \cup \xi^{\dot{C}L}$ and $\sigma \cup \xi^{\dot{C}L} \models \neg b$. This leads to a contradiction. Therefore this case cannot occur.

(2) Rule 25.1.r has been applied. The proof is similar to the proof of the previous case.

Condition 4: We assume $E \Vdash \langle b \to (p \,[\!]\, q), \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma'\rangle$ for some $E, \sigma, t, \rho, k_1, \sigma'$, which means that either Rule 21 or Rule 22 has been applied necessarily. Then

we can distinguish two cases:

(1) Rule 21 has been applied. Then, $E \Vdash \langle p \parallel q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_{pq}, \sigma'\rangle$ for some k_{pq} such that $k_1 \equiv b \to k_{pq}$ and $\forall_{s\in[0,t]}\, \rho(s) \models b$. For $E \Vdash \langle p \parallel q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_{pq}, \sigma'\rangle$, we get $E \Vdash \langle p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_p, \sigma'\rangle$ and $E \Vdash \langle q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_q, \sigma'\rangle$ for some k_p, k_q such that $k_{pq} \equiv k_p \parallel k_q$ (using Rule 26). Applying Rule 21, we obtain $E \Vdash \langle b \to p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle b \to k_p, \sigma'\rangle$ and $E \Vdash \langle b \to q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle b \to k_q, \sigma'\rangle$. According to Rule 26, we have $E \Vdash \langle b \to p \parallel b \to q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle b \to k_p \parallel b \to k_q, \sigma'\rangle$. Note that $k_1 \equiv b \to k_p \parallel k_q$. Take $k_2 \equiv b \to k_p \parallel b \to k_q$ and observe that $(k_1, k_2) \in R$.

(2) Rule 22 has been applied. Then, $k_1 \equiv b \to (p \parallel q)$ and $\sigma' = \rho_\sigma(t)$, $\rho \in \Omega_{\sigma Et}, \forall_{s\in(0,t)}\, \rho(s) \models \neg b, \rho(0) \models b \Rightarrow E \Vdash \langle p \parallel q, \sigma\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle z, \sigma''\rangle$ for some z, σ'' and $\rho(t) \models b \Rightarrow E \Vdash \langle p \parallel q, \rho_\sigma(t)\rangle \overset{\rho(t)}{\rightsquigarrow}$. From $\rho(0) \models b \Rightarrow E \Vdash \langle p \parallel q, \sigma\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle z, \sigma''\rangle$, we can also have $\rho(0) \models b \Rightarrow E \Vdash \langle p, \sigma\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle p_z, \sigma''\rangle$ for some p_z, and $\rho(0) \models b \Rightarrow E \Vdash \langle q, \sigma\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle q_z, \sigma''\rangle$ for some q_z (see also Rule 26). From $\rho(t) \models b \Rightarrow E \Vdash \langle p \parallel q, \rho_\sigma(t)\rangle \overset{\rho(t)}{\rightsquigarrow}$, we also get $\rho(t) \models b \Rightarrow E \Vdash \langle p, \rho_\sigma(t)\rangle \overset{\rho(t)}{\rightsquigarrow}$ and $\rho(t) \models b \Rightarrow E \Vdash \langle q, \rho_\sigma(t)\rangle \overset{\rho(t)}{\rightsquigarrow}$ (see also Rule 27). Using Rule 22, we obtain $E \Vdash \langle b \to p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle b \to p, \rho_\sigma(t)\rangle$ and $E \Vdash \langle b \to q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle b \to q, \rho_\sigma(t)\rangle$. According to Rule 26, we obtain $E \Vdash \langle b \to p \parallel b \to q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle b \to p \parallel b \to q, \rho_\sigma(t)\rangle$. Take $k_2 \equiv b \to p \parallel b \to q$ and observe that $(k_1, k_2) \in R$.

Condition 5: We assume $E \Vdash \langle b \to p \parallel b \to q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma'\rangle$ for some E, σ, t, ρ, k_1, σ', which means that Rule 26 has been applied necessarily. Then $E \Vdash \langle b \to p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_p, \sigma'\rangle$, and $E \Vdash \langle b \to q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_q, \sigma'\rangle$ for some k_p, k_q such that $k_1 \equiv k_p \parallel k_q$. For $E \Vdash \langle b \to p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_p, \sigma'\rangle$, and $E \Vdash \langle b \to q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_q, \sigma'\rangle$, four cases can be distinguished:

(1) Rule 21 has been applied for both. Then, $E \Vdash \langle p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k'_p, \sigma'\rangle$, $E \Vdash \langle q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k'_q, \sigma'\rangle$ for some k'_p, k'_q such that $k_p \equiv b \to k'_p$, $k_q \equiv b \to k'_q$, and $\forall_{s\in[0,t]}\, \rho(s) \models b$. Using Rule 26, we obtain $E \Vdash \langle p \parallel q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k'_p \parallel k'_q, \sigma'\rangle$. Applying Rule 21, we get $E \Vdash \langle b \to (p \parallel q), \sigma\rangle \overset{t,\rho}{\longmapsto} \langle b \to k'_p \parallel k'_q, \sigma'\rangle$. Note that $k_1 \equiv b \to k'_p \parallel b \to k'_q$. Take $k_2 \equiv b \to k'_p \parallel k'_q$ and observe that $(k_2, k_1) \in R$.

(2) Rule 22 has been applied for both. Then, $k_p \equiv b \to p, k_q \equiv b \to q$ and $\sigma' = \rho_\sigma(t)$, $\rho \in \Omega_{\sigma Et}$, $\forall_{s\in(0,t)}\, \rho(s) \models \neg b$, $\rho(0) \models b \Rightarrow E \Vdash \langle p, \sigma\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle p_z, \sigma''\rangle, \rho(0) \models b \Rightarrow E \Vdash \langle q, \sigma\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle q_z, \sigma'''\rangle$, for some p_z, q_z, σ'', σ''', $\rho(t) \models b \Rightarrow E \Vdash \langle p, \rho_\sigma(t)\rangle \overset{\rho(t)}{\rightsquigarrow}$ and $\rho(t) \models b \Rightarrow E \Vdash \langle q, \rho_\sigma(t)\rangle \overset{\rho(t)}{\rightsquigarrow}$. From $\rho(0) \models b \Rightarrow E \Vdash \langle p, \sigma\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle p_z, \sigma''\rangle$, and $\rho(0) \models b \Rightarrow E \Vdash \langle q, \sigma\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto}$

$\langle q_z, \sigma''' \rangle$, by Lemma 1 we know that $\sigma'' = \sigma''' = \rho_\sigma(0)$, we get $\rho(0) \models b \Rightarrow E \Vdash \langle p \,[\!]\, q, \sigma \rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle z', \sigma'' \rangle$ for some z' (see also Rule 26). From $\rho(t) \models b \Rightarrow E \Vdash \langle p, \rho_\sigma(t) \rangle \overset{\rho(t)}{\rightsquigarrow}$ and $\rho(t) \models b \Rightarrow E \Vdash \langle q, \rho_\sigma(t) \rangle \overset{\rho(t)}{\rightsquigarrow}$, we get $\rho(t) \models b \Rightarrow E \Vdash \langle p \,[\!]\, q, \rho_\sigma(t) \rangle \overset{\rho(t)}{\rightsquigarrow}$ (see also Rule 27). Using Rule 22, we obtain $E \Vdash \langle b \to (p \,[\!]\, q), \sigma \rangle \overset{t,\rho}{\longmapsto} \langle b \to (p \,[\!]\, q), \rho_\sigma(t) \rangle$. Notice that $k_1 \equiv b \to p \,[\!]\, b \to q$. Take $k_2 \equiv b \to (p \,[\!]\, q)$ and observe that $(k_2, k_1) \in R$.

(3) Rule 21 has been applied for $E \Vdash \langle b \to p, \sigma \rangle \overset{t,\rho}{\longmapsto} \langle k_p, \sigma' \rangle$, and Rule 22 has been applied for $E \Vdash \langle b \to q, \sigma \rangle \overset{t,\rho}{\longmapsto} \langle k_q, \sigma' \rangle$. Then, $E \Vdash \langle p, \sigma \rangle \overset{t,\rho}{\longmapsto} \langle k'_p, \sigma' \rangle$ for some k'_p such that $k_p \equiv b \to k'_p$, $\sigma' = \rho_\sigma(t)$, $\forall_{s\in[0,t]}\, \rho(s) \models b$, $\rho \in \Omega_{\sigma Et}$, $\forall_{s\in(0,t)}\, \rho(s) \models \neg b$, $\rho(0) \models b \Rightarrow E \Vdash \langle q, \sigma \rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle q_z, \sigma'' \rangle$, for some q_z, σ'', $\rho(t) \models b \Rightarrow E \Vdash \langle q, \rho_\sigma(t) \rangle \overset{\rho(t)}{\rightsquigarrow}$, $k_q \equiv b \to q$, and $k_1 \equiv b \to k'_p \,[\!]\, b \to q$. From $\forall_{s\in[0,t]}\, \rho(s) \models b$, and $\forall_{s\in(0,t)}\, \rho(s) \models \neg b$, this leads to a contradiction, unless $t = 0$. Hence, $t = 0$. Then we consider only the case in which $t = 0$. From $E \Vdash \langle p, \sigma \rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle k'_p, \sigma' \rangle$, $\rho(0) \models b$, and $\rho(0) \models b \Rightarrow E \Vdash \langle q, \sigma \rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle q_z, \sigma'' \rangle$, it is not hard to see that we get $\rho(0) \models b \Rightarrow E \Vdash \langle p \,[\!]\, q, \sigma \rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle k_z, \sigma''' \rangle$ for some k_z, σ'''. We know that $\sigma' = \sigma'' = \sigma''' = \rho_\sigma(0)$ (see also Rule 26 and Lemma 1). Since $\rho \in \Omega_{\sigma Et}$, we have $\sigma = \rho_\sigma(0)$. Also, from $E \Vdash \langle p, \rho_\sigma(0) \rangle \overset{0,\rho}{\longmapsto} \langle k'_p, \sigma' \rangle$, we have $E \Vdash \langle p, \rho_\sigma(0) \rangle \overset{\rho(0)}{\rightsquigarrow}$ (by Lemma 3). Using Rule 26, we $\rho(0) \models b \Rightarrow E \Vdash \langle p \,[\!]\, q, \rho_\sigma(0) \rangle \overset{\rho(0)}{\rightsquigarrow}$. Applying Rule 22, we obtain $E \Vdash \langle b \to (p \,[\!]\, q), \sigma \rangle \overset{0,\rho}{\longmapsto} \langle b \to (p \,[\!]\, q), \rho_\sigma(0) \rangle$. Take $k_2 \equiv b \to (p \,[\!]\, q)$ and observe that $(k_2, k_1) \in R$.

(4) Rule 21 has been applied for $E \Vdash \langle b \to q, \sigma \rangle \overset{t,\rho}{\longmapsto} \langle k_q, \sigma' \rangle$, and Rule 22 has been applied for $E \Vdash \langle b \to p, \sigma \rangle \overset{t,\rho}{\longmapsto} \langle k_p, \sigma' \rangle$. The proof is similar to the previous case.

Condition 6: First, we assume $(C, J, L, H, R) \Vdash \langle b \to (p \,[\!]\, q), \sigma \rangle \overset{\xi}{\rightsquigarrow}$ for some C, J, L, H, R, σ, ξ, which means that Rule 23 or Rule 24 has been applied necessarily. Then, we distinguish two cases:

(1) Rule 23 has been applied. Then $(C, J, L, H, R) \Vdash \langle p \,[\!]\, q, \sigma \rangle \overset{\xi}{\rightsquigarrow}$ and $\xi \models b$. Using Rule 27, we have $(C, J, L, H, R) \Vdash \langle p, \sigma \rangle \overset{\xi}{\rightsquigarrow}$ and $(C, J, L, H, R) \Vdash \langle q, \sigma \rangle \overset{\xi}{\rightsquigarrow}$. According to Rule 23, we obtain $(C, J, L, H, R) \Vdash \langle b \to p, \sigma \rangle \overset{\xi}{\rightsquigarrow}$ and $(C, J, L, H, R) \Vdash \langle b \to q, \sigma \rangle \overset{\xi}{\rightsquigarrow}$. Applying Rule 27, we get $(C, J, L, H, R) \Vdash \langle b \to p \,[\!]\, b \to q, \sigma \rangle \overset{\xi}{\rightsquigarrow}$.

(2) Rule 24 has been applied. Then $\xi = \sigma \cup \xi^{\dot{C}L}$ for some $\sigma \cup \xi^{\dot{C}L}$ and $\sigma \cup \xi^{\dot{C}L} \models \neg b$. Using Rule 24, we can have $(C, J, L, H, R) \Vdash \langle b \to p, \sigma \rangle \overset{\sigma\cup\xi^{\dot{C}L}}{\rightsquigarrow}$ and $(C, J, L, H, R) \Vdash \langle b \to q, \sigma \rangle \overset{\sigma\cup\xi^{\dot{C}L}}{\rightsquigarrow}$. Applying Rule 27, we get that $(C, J, L, H, R) \Vdash \langle b \to p \,[\!]\, b \to q, \sigma \rangle \overset{\sigma\cup\xi^{\dot{C}L}}{\rightsquigarrow}$.

Second, we assume $(C, J, L, H, R) \Vdash \langle b \to p \mathbin{[\!]} b \to q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some C, J, L, H, R, σ, ξ, which means that Rule 27 has been applied necessarily. Then, $(C, J, L, H, R) \Vdash \langle b \to p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and $(C, J, L, H, R) \Vdash \langle b \to q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. For (C, J, L, H, R) $\langle b \to p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and $(C, J, L, H, R) \Vdash \langle b \to q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, four cases can be distinguished:

(1) Rule 23 has been applied for both. Then, we have $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and $\xi \models b$. According to Rule 27, we obtain $(C, J, L, H, R) \Vdash \langle p \mathbin{[\!]} q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 23, we get $(C, J, L, H, R) \Vdash \langle b \to (p \mathbin{[\!]} q), \sigma\rangle \overset{\xi}{\rightsquigarrow}$.

(2) Rule 24 has been applied for both. Then $\xi \equiv \sigma \cup \xi^{\dot{C}L}$ for some $\sigma \cup \xi^{\dot{C}L}$ and $\sigma \cup \xi^{\dot{C}L} \models \neg b$. According to Rule 24, we can have $(C, J, L, H, R) \Vdash \langle b \to (p \mathbin{[\!]} q), \sigma\rangle \overset{\sigma\cup\xi^{\dot{C}L}}{\rightsquigarrow}$.

(3) Rule 23 has been applied for $(C, J, L, H, R) \Vdash \langle b \to p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and Rule 24 has been applied for $(C, J, L, H, R) \Vdash \langle b \to q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Then, $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $\xi \models b$, and $\xi = \sigma \cup \xi^{\dot{C}L}$, and $\sigma \cup \xi^{\dot{C}L} \models \neg b$. This leads to a contradiction. Therefore, Rule 23 cannot have been applied for $(C, J, L, H, R) \Vdash \langle b \to p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and Rule 24 cannot have been applied for $(C, J, L, H, R) \Vdash \langle b \to q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$.

(4) Rule 23 has been applied for $(C, J, L, H, R) \Vdash \langle b \to q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and Rule 24 has been applied for $(C, J, L, H, R) \Vdash \langle b \to p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. The proof is similar to the previous case.

B.4 Properties of sequential composition

Lemma 25 (Left-zero element for sequential comp.) *For every closed process term p we have*

$$\delta;\ p \leftrightarrow \delta.$$

PROOF. Let $R = \{(\delta;\ p, \delta) \mid p \in P\}$. Since there are no action transition rules and time transition rules defined for δ, and therefore also not for $\delta;\ p$, the conditions 1 – 5 hold trivially.

Condition 6: First, we assume $(C, J, L, H, R) \Vdash \langle \delta;\ p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some C, J, L, H, R, σ, ξ, which means that Rule 19 has been applied necessarily. Then, $(C, J, L, H, R) \Vdash \langle \delta, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Second, we assume $(C, J, L, H, R) \Vdash \langle \delta, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 19, we obtain $(C, J, L, H, R) \Vdash \langle \delta;\ p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$.

Lemma 26 (Associativity of sequential composition) *For every closed process terms p, q and r we have*

$$(p;\ q);\ r \leftrightarrows p;\ (q;\ r).$$

PROOF. Let $R = \{((p;\ q);\ r, p;\ (q;\ r)) \mid p, q, r \in P\} \cup \{(i_d, i_d) \mid i_d \in P\}$. The proofs of conditions 4 and 5 are similar to the proofs of conditions 2 and 3 (except Rule 16 has not been applied, because no χ process can transform to terminated process by means of time transitions) since the deduction rules for non-terminating action transitions and time transitions of ; are similar.

Condition 1: Since there are no termination transitions defined for the transition $E \Vdash \langle (p;\ q);\ r, \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle \surd, \sigma'\rangle$ and $E \Vdash \langle p;\ (q;\ r), \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle \surd, \sigma'\rangle$, condition 1 holds trivially.

Condition 2: We assume $E \Vdash \langle (p;\ q);\ r, \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle k_1, \sigma'\rangle$ for some E, σ, ξ, a, ξ', k_1, σ', which means that either Rule 16 or Rule 17 has been applied necessarily. Hence, we distinguish two cases:

(1) Rule 16 has been applied. Then $E \Vdash \langle p;\ q, \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle \surd, \sigma'\rangle$. This leads to a contradiction as there is no deduction rule that allows a sequential composition to perform a termination transition. Hence, this case cannot occur.
(2) Rule 17 has been applied. Then, $E \Vdash \langle p;\ q, \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle k_1', \sigma'\rangle$ for some k_1' such that $k_1 \equiv k_1';\ r$. We distinguish two cases for $E \Vdash \langle p;\ q, \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle k_1', \sigma'\rangle$:
   (a) Rule 16 has been applied. Then, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle \surd, \sigma'\rangle$, $E \Vdash \langle q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$ and $k_1' \equiv q$. According to Rule 19, we have $E \Vdash \langle q;\ r, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. Using Rule 16, we have $E \Vdash \langle p;\ (q;\ r), \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle q;\ r, \sigma'\rangle$. Note that $k_1 \equiv q;\ r$. Take $k_2 \equiv q;\ r$ and observe that $(k_1, k_2) \in R$.
   (b) Rule 17 has been applied. Then, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle k_p, \sigma'\rangle$ for some k_p such that $k_1' \equiv k_p;\ q$. Using Rule 17 we obtain $E \Vdash \langle p;\ (q;\ r), \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle k_p;\ (q;\ r), \sigma'\rangle$. Note that $k_1 \equiv (k_p;\ q);\ r$. Take $k_2 \equiv k_p;\ (q;\ r)$ and observe that $(k_1, k_2) \in R$.

Condition 3: We assume $E \Vdash \langle p;\ (q;\ r), \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle k_1, \sigma'\rangle$ for some E, σ, ξ, a, ξ', k_1, σ', which means that either Rule 16 or Rule 17 has been applied necessarily. Hence, we distinguish two cases:

(1) Rule 16 has been applied. Then, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle \surd, \sigma'\rangle$, $E \Vdash \langle q;\ r, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$ and $k_1 \equiv q;\ r$. According to Rule 19, we have $E \Vdash \langle q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. Using Rule 16, we obtain $E \Vdash \langle p;\ q, \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle q, \sigma'\rangle$. Using Rule 17, we obtain $E \Vdash$

$\langle (p;\ q);\ r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle q;\ r, \sigma'\rangle$ and observe that $(k_1, k_1) \in R$.

(2) Rule 17 has been applied. Then, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_p, \sigma'\rangle$ for some k_p such that $k_1 \equiv k_p;\ (q;\ r)$. Using Rule 17, we obtain $E \Vdash \langle p;\ q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_p;\ q, \sigma'\rangle$. Again, using Rule 17, we obtain $E \Vdash \langle (p;\ q);\ r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle (k_p;\ q);\ r, \sigma'\rangle$. Take $k_2 \equiv (k_p;\ q);\ r$ and observe that $(k_2, k_1) \in R$.

Condition 6: First, we assume $E \Vdash \langle (p;\ q);\ r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some E, σ, ξ, which means that Rule 19 has applied necessarily. Then $E \Vdash \langle p;\ q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Again, due to Rule 19, we get $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 19, we obtain $E \Vdash \langle p;\ (q;\ r), \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Second, we assume $E \Vdash \langle p;\ (q;\ r), \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some E, σ, ξ, which means that Rule 19 has applied necessarily. Then $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Again, due to Rule 19, we get $E \Vdash \langle p;\ q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 19, we obtain $E \Vdash \langle (p;\ q);\ r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$.

Lemma 27 (Distribution of sequential over alternative comp.) *For p, q and r arbitrary closed process terms we have*

$$(p \between q);\ r \leftrightarrows p;\ r \between q;\ r.$$

PROOF. Let $R = \{((p \between q);\ r, p;\ r \between q;\ r) \mid p, q, r \in P\} \cup \{(i_d, i_d) \mid i_d \in P\}$.

Condition 1: Since there no action transition rules defined for any closed process term k_1 and k_2 such that $E \Vdash \langle k_1;\ k_2, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$, condition 1 holds trivially.

Condition 2: We assume $E \Vdash \langle (p \between q);\ r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ for some E, σ, ξ, a, ξ', k_1, σ'; which means that either Rule 16 or Rule 17 has been applied necessarily. Hence, we can distinguish two cases:

(1) Rule 16 has been applied. Then, $E \Vdash \langle p \between q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$, and $E \Vdash \langle r, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. For $E \Vdash \langle p \between q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ we again distinguish two cases:
   (a) Rule 25.1.l has been applied. Then, $k_1 \equiv r$, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ and $E \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 16, we get $E \Vdash \langle p;\ r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle r, \sigma'\rangle$. Due to Rule 19, we have $E \Vdash \langle q;\ r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. According to Rule 25.2.l, we have $E \Vdash \langle p;\ r \between q;\ r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle r, \sigma'\rangle$ and observe that $(r, r) \in R$.
   (b) Rule 25.1.r has been applied. The proof is similar to the proof of the previous case.

(2) Rule 17 has been applied. Then, $E \Vdash \langle p \between q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1', \sigma'\rangle$ for some k_1' such that $k_1 \equiv k_1';\ r$. For $E \Vdash \langle p \between q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1', \sigma'\rangle$ we can further distinguish two cases:

(a) Rule 25.2.l has been applied. Then, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1', \sigma'\rangle$ and $E \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 17, we get $E \Vdash \langle p;\ r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1';\ r, \sigma'\rangle$. Using Rule 19, we get $\langle q;\ r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. According to Rule 25.2.l, we have $E \Vdash \langle p;\ r \parallel q;\ r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1';\ r, \sigma'\rangle$. Take $k_2 \equiv k_1';\ r$ and observe that $(k_1, k_2) \in R$.

(b) Rule 25.2.r has been applied. The proof is similar to the proof of the previous case.

Condition 3: We assume $E \Vdash \langle p;\ r \parallel q;\ r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ for some E, σ, ξ, a, ξ', k_1, σ'; which means that either Rule 25.2.l or Rule 25.2.r has been applied necessarily. Hence, we can distinguish two cases:

(1) Rule 25.2.l has been applied. Then, $E \Vdash \langle p;\ r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ and $E \Vdash \langle q;\ r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. For $E \Vdash \langle q;\ r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, we also get $E \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ using Rule 19. For $E \Vdash \langle p;\ r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ we again distinguish two cases:

(a) Rule 16 has been applied. Then, $k_1 \equiv r$, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$, and $\langle r, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. Applying Rule 25.1.l, we get $E \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$. According to Rule 16, we have $E \Vdash \langle (p \parallel q);\ r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle r, \sigma'\rangle$ and observe that $(r, r) \in R$.

(b) Rule 17 has been applied. Then, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_p, \sigma'\rangle$ for some k_p such that $k_1 \equiv k_p;\ r$. Using Rule 25.2.l, we get $E \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_p, \sigma'\rangle$. According to Rule 17, we have $E \Vdash \langle (p \parallel q);\ r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_p;\ r, \sigma'\rangle$ and observe that $(k_1, k_1) \in R$.

(2) Rule 25.2.r has been applied. The proof is similar to the proof of the previous case.

Condition 4: We assume $E \Vdash \langle (p \parallel q);\ r, \sigma\rangle \xmapsto{t,\rho} \langle k_1, \sigma'\rangle$ for some E, σ, t, ρ, k_1, σ'; which means Rule 18 has been applied necessarily. Then, $E \Vdash \langle p \parallel q, \sigma\rangle \xmapsto{t,\rho} \langle k_{pq}, \sigma'\rangle$ for some k_{pq} such that $k_1 \equiv k_{pq};\ r$. For $E \Vdash \langle p \parallel q, \sigma\rangle \xmapsto{t,\rho} \langle k_{pq}, \sigma'\rangle$, Rule 26 has been applied necessarily. Then, $E \Vdash \langle p, \sigma\rangle \xmapsto{t,\rho} \langle k_p, \sigma'\rangle$ and $E \Vdash \langle q, \sigma\rangle \xmapsto{t,\rho} \langle k_q, \sigma'\rangle$ for some k_p, k_q such that $k_{pq} \equiv k_p \parallel k_q$. Using Rule 18, we obtain $E \Vdash \langle p;\ r, \sigma\rangle \xmapsto{t,\rho} \langle k_p;\ r, \sigma'\rangle$ and $E \Vdash \langle q;\ r, \sigma\rangle \xmapsto{t,\rho} \langle k_q;\ r, \sigma'\rangle$. According to Rule 26 we obtain $E \Vdash \langle p;\ r \parallel q;\ r, \sigma\rangle \xmapsto{t,\rho} \langle k_p;\ r \parallel k_q;\ r, \sigma'\rangle$. Note that $k_1 \equiv (k_p \parallel k_q);\ r$. Take $k_2 \equiv k_p;\ r \parallel k_q;\ r$ and observe that $(k_1, k_2) \in R$.

Condition 5: We assume $E \Vdash \langle p;\ r \parallel q;\ r, \sigma\rangle \xmapsto{t,\rho} \langle k_1, \sigma'\rangle$ for some E, σ, t, ρ, k_1, σ'; which means Rule 26 has been applied necessarily. Then, $E \Vdash \langle p;\ r, \sigma\rangle \xmapsto{t,\rho} \langle k_{pr}, \sigma'\rangle$ and $E \Vdash \langle q;\ r, \sigma\rangle \xmapsto{t,\rho} \langle k_{qr}, \sigma'\rangle$ for some k_{pr}, k_{qr} such that $k_1 \equiv k_{pr} \parallel k_{qr}$. For $E \Vdash \langle p;\ r, \sigma\rangle \xmapsto{t,\rho} \langle k_{pr}, \sigma'\rangle$ and $E \Vdash \langle q;\ r, \sigma\rangle \xmapsto{t,\rho} \langle k_{qr}, \sigma'\rangle$, Rule 18 has been

applied to both. Then, $E \Vdash \langle p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_p, \sigma'\rangle$ and $E \Vdash \langle q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_q, \sigma'\rangle$ for some k_p, k_q such that $k_{pr} \equiv k_p; r$ and $k_{qr} \equiv k_q; r$. Using Rule 26 we then obtain $E \Vdash \langle p \parallel q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_p \parallel k_q, \sigma'\rangle$. Applying Rule 19, we get $E \Vdash \langle (p \parallel q); r, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle (k_p \parallel k_q); r, \sigma'\rangle$. Note that $k_1 \equiv k_p; r \parallel k_q; r$. Take $k_2 \equiv (k_p \parallel k_q); r$ and observe that $(k_2, k_1) \in R$.

Condition 6: First, we assume $E \Vdash \langle (p \parallel q); r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some E, σ, ξ; which means Rule 19 has been applied necessarily. Then, $E \Vdash \langle p \parallel q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 27, we have $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and $E \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Applying Rule 19, we get $E \Vdash \langle p;\ r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and $E \Vdash \langle q;\ r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. According to Rule 27, we get $E \Vdash \langle p;\ r \parallel q;\ r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Second, we assume $E \Vdash \langle p;\ r \parallel q;\ r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some E, σ, ξ; which means Rule 27 has been applied necessarily. Then, $E \Vdash \langle p;\ r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and $E \Vdash \langle q;\ r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Due to Rule 19, we obtain $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and $E \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 27, we get $E \Vdash \langle p \parallel q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Applying Rule 19, we have $E \Vdash \langle (p \parallel q);\ r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$.

Lemma 28 *For arbitrary closed process terms p and q and arbitrary guard b we have*

$$b \to (p;\ q) \leftrightarrows b \to p;\ q.$$

PROOF. Let $R = \{(b \to (p;\ q), b \to p;\ q) \mid p, q \in P, \text{guard } b\} \cup \{(i_d, i_d) \mid i_d \in P\}$.

Condition 1: We assume $E \Vdash \langle b \to (p;\ q), \sigma\rangle \overset{\xi,a,\xi'}{\longrightarrow} \langle \surd, \sigma'\rangle$ for some E, σ, ξ, a, ξ', σ', which means that Rule 20.1 has been applied necessarily. Then, $E \Vdash \langle p;\ q, \sigma\rangle \overset{\xi,a,\xi'}{\longrightarrow} \langle \surd, \sigma'\rangle$ and $\xi \models b$. This leads to a contradiction as there is no deduction rule that allows a sequential composition to perform a termination transition. Second, we assume $E \Vdash \langle b \to p;\ q, \sigma\rangle \overset{\xi,a,\xi'}{\longrightarrow} \langle \surd, \sigma'\rangle$ for some E, σ, ξ, a, ξ', σ'. This also leads to a contradiction as there is no deduction rule that allows a sequential composition to perform a termination transition. Thus, condition 1 holds trivially.

Condition 2: We assume $E \Vdash \langle b \to (p;\ q), \sigma\rangle \overset{\xi,a,\xi'}{\longrightarrow} \langle k_1, \sigma'\rangle$ for some E, σ, ξ, a, ξ', k_1, σ', which means that Rule 20.2 has been applied necessarily. Then, we have $E \Vdash \langle p;\ q, \sigma\rangle \overset{\xi,a,\xi'}{\longrightarrow} \langle k_1, \sigma'\rangle$, and $\xi \models b$. For $E \Vdash \langle p;\ q, \sigma\rangle \overset{\xi,a,\xi'}{\longrightarrow} \langle k_1, \sigma'\rangle$, two cases can be distinguished:

(1) Rule 16 has been applied. Then, $k_1 \equiv q$, $E \Vdash \langle p, \sigma\rangle \overset{\xi,a,\xi'}{\longrightarrow} \langle \surd, \sigma'\rangle$ and $E \Vdash \langle q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. Using Rule 20.1 we have $E \Vdash \langle b \to p, \sigma\rangle \overset{\xi,a,\xi'}{\longrightarrow} \langle \surd, \sigma'\rangle$. Applying Rule 16 we have $E \Vdash \langle b \to p;\ q, \sigma\rangle \overset{\xi,a,\xi'}{\longrightarrow} \langle q, \sigma'\rangle$ and observe that $(q, q) \in R$.

(2) Rule 17 has been applied. Then, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_p, \sigma'\rangle$ for some k_q such that $k_1 \equiv k_p;\ q$. Using Rule 20.2 we have $E \Vdash \langle b \to p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_p, \sigma'\rangle$, and using Rule 17 we have $E \Vdash \langle b \to p;\ q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_p;\ q, \sigma'\rangle$ and observe that $(k_1, k_1) \in R$.

Condition 3: We assume $E \Vdash \langle b \to p;\ q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ for some E, σ, ξ, a, ξ', k_1, σ', which means that Rule 16 or Rule 17 has been applied necessarily. Then, we distinguish two cases:

(1) Rule 16 has been applied. Then, $k_1 \equiv q$, $E \Vdash \langle b \to p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$ and $E \Vdash \langle q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. According to Rule 20.1 we have $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$ and $\xi \models b$. Applying Rule 16 we have $E \Vdash \langle p;\ q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle q, \sigma'\rangle$. Using Rule 20.2 we get $E \Vdash \langle b \to (p;\ q), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle q, \sigma'\rangle$ and observe that $(q, q) \in R$.
(2) Rule 17 has been applied. Then $E \Vdash \langle b \to p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_p, \sigma'\rangle$ for some k_p such that $k_1 \equiv k_p;\ q$. Using Rule 20.2, we obtain $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_p, \sigma'\rangle$ and $\xi \models b$. Using Rule 17, we get $E \Vdash \langle p;\ q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_p;\ q, \sigma'\rangle$. Applying Rule 20.2, we have $E \Vdash \langle b \to (p;\ q), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_p;\ q, \sigma'\rangle$ and observe that $(k_1, k_1) \in R$.

Condition 4: We assume $E \Vdash \langle b \to (p;\ q), \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma'\rangle$ for some $E, \sigma, t, \rho, k_1, \sigma'$, which means that either Rule 21 or Rule 22 has been applied necessarily. Then we can distinguish two cases:

(1) Rule 21 has been applied. Then, $E \Vdash \langle p;\ q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1', \sigma'\rangle$ and $\forall_{s\in[0,t]}\, \rho(t) \models b$ such that $k_1 \equiv b \to k_1'$. For $E \Vdash \langle p;\ q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1', \sigma'\rangle$, we get $E \Vdash \langle p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_p, \sigma'\rangle$ for some k_p such that $k_1' \equiv k_p;\ q$ using Rule 18. Applying Rule 21, we get $E \Vdash \langle b \to p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle b \to k_p, \sigma'\rangle$. According to Rule 18, we obtain $E \Vdash \langle b \to p;\ q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle (b \to k_p);\ q, \sigma'\rangle$. Note that $k_1 \equiv b \to (k_p;\ q)$. Take $k_2 \equiv (b \to k_p);\ q$ and observe that $(k_1, k_2) \in R$.
(2) Rule 22 has been applied. Then, $k_1 \equiv b \to (p;\ q)$, $\sigma' = \rho_\sigma(t)$, $\rho \in \Omega_{\sigma E t}$, $\forall_{s\in(0,t)}\ \rho(s) \models \neg b$, $\rho(0) \models b \Rightarrow E \Vdash \langle p;\ q, \sigma\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle p', \sigma''\rangle$, for some p', σ'', $\rho(t) \models b \Rightarrow E \Vdash \langle p;\ q, \rho_\sigma(t)\rangle \overset{\rho(t)}{\rightsquigarrow}$. For $\rho(0) \models b \Rightarrow E \Vdash \langle p;\ q, \sigma\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle p', \sigma''\rangle$, we get $\rho(0) \models b \Rightarrow E \Vdash \langle p, \sigma\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle k_p', \sigma''\rangle$, for some k_p' (see also Rule 18). For $\rho(t) \models b \Rightarrow E \Vdash \langle p;\ q, \rho_\sigma(t)\rangle \overset{\rho(t)}{\rightsquigarrow}$, we get $\rho(t) \models b \Rightarrow E \Vdash \langle p, \rho_\sigma(t)\rangle \overset{\rho(t)}{\rightsquigarrow}$ (see also Rule 19). Using Rule 22, we obtain $E \Vdash \langle b \to p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle b \to p, \rho_\sigma(t)\rangle$. Using Rule 18, we obtain $E \Vdash \langle b \to p;\ q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle b \to p;\ q, \rho_\sigma(t)\rangle$. Take $k_2 \equiv b \to p;\ q$ and observe that $(k_1, k_2) \in R$.

Condition 5: We assume $E \Vdash \langle b \to p;\ q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1, \sigma'\rangle$ for some $E, \sigma, t, \rho, k_1, \sigma'$,

which means that Rule 18 has been applied necessarily. Then $E \Vdash \langle b \to p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1', \sigma'\rangle$ for some k_1' such that $k_1 \equiv k_1';\ q$. For $E \Vdash \langle b \to p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_1', \sigma'\rangle$, two cases can be distinguished:

(1) Rule 21 has been applied. Then, $E \Vdash \langle p, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_p, \sigma'\rangle$ and $\forall_{s\in[0,t]}\, \rho(t) \models b$ for some k_p such that $k_1' \equiv b \to k_p$. Using Rule 18, we get $E \Vdash \langle p;\ q, \sigma\rangle \overset{t,\rho}{\longmapsto} \langle k_p;\ q, \sigma'\rangle$. According to Rule 21, we have $E \Vdash \langle b \to (p;\ q), \sigma\rangle \overset{t,\rho}{\longmapsto} \langle b \to (k_p;\ q), \sigma'\rangle$. Note that $k_1 \equiv b \to k_p;\ q$. Take $k_2 \equiv b \to (k_p;\ q)$ and observe that $(k_2, k_1) \in R$.
(2) Rule 22 has been applied. Then, $k_1' \equiv b \to p$, $\sigma' = \rho_\sigma(t)$, $\rho \in \Omega_{\sigma E t}$, $\forall_{s\in(0,t)}$ $\rho(s) \models \neg b$, $\rho(0) \models b \Rightarrow E \Vdash \langle p, \sigma\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle p', \sigma''\rangle$, for some p', σ'', and $\rho(t) \models b \Rightarrow E \Vdash \langle p, \rho_\sigma(t)\rangle \overset{\rho(t)}{\rightsquigarrow}$. From $\rho(0) \models b \Rightarrow E \Vdash \langle p, \sigma\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle p', \sigma''\rangle$, we get $\rho(0) \models b \Rightarrow E \Vdash \langle p;\ q, \sigma\rangle \overset{0,\rho\restriction\{0\}}{\longmapsto} \langle k_p', \sigma''\rangle$ for some k_p'. From $\rho(t) \models b \Rightarrow E \Vdash \langle p, \rho_\sigma(t)\rangle \overset{\rho(t)}{\rightsquigarrow}$, we get $\rho(t) \models b \Rightarrow E \Vdash \langle p;\ q, \rho_\sigma(t)\rangle \overset{\rho(t)}{\rightsquigarrow}$ using Rule 19. Applying Rule 22, we obtain $E \Vdash \langle b \to (p;\ q), \sigma\rangle \overset{t,\rho}{\longmapsto} \langle b \to (p;\ q), \rho_\sigma(t)\rangle$. Note that $k_1 \equiv b \to p;\ q$. Take $k_2 \equiv b \to (p;\ q)$ and observe that $(k_2, k_1) \in R$.

Condition 6: First, we assume $(C, J, L, H, R) \Vdash \langle b \to (p;\ q), \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some C, J, L, H, R, σ, ξ, which means that either Rule 23 or Rule 24 has been applied necessarily. Then we can distinguish two cases:

(1) Rule 23 has been applied. Then, $(C, J, L, H, R) \Vdash \langle p;\ q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and $\xi \models b$. Rule 19, we obtain $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Applying Rule 23, we get $(C, J, L, H, R) \Vdash \langle b \to p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Again, due to Rule 19, we obtain $(C, J, L, H, R) \Vdash \langle b \to p;\ q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$.
(2) Rule 24 has been applied. Then, $\xi = \sigma \cup \xi^{\dot{C}L}$ for some $\sigma \cup \xi^{\dot{C}L}$ and $\sigma \cup \xi^{\dot{C}L} \models \neg b$. Using Rule 24 we get $(C, J, L, H, R) \Vdash \langle b \to p, \sigma\rangle \overset{\sigma\cup\xi^{\dot{C}L}}{\rightsquigarrow}$. Applying Rule 19, we obtain $(C, J, L, H, R) \Vdash \langle b \to p;\ q, \sigma\rangle \overset{\sigma\cup\xi^{\dot{C}L}}{\rightsquigarrow}$.

Second, we assume $(C, J, L, H, R) \Vdash \langle b \to p;\ q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some C, J, L, H, R, σ, ξ, which means that Rule 19 has been applied necessarily. Then $(C, J, L, H, R) \Vdash \langle b \to p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. For this, we can distinguish two cases:

(1) Rule 23 has been applied. Then, $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and $\xi \models b$. Using Rule 19, we obtain $(C, J, L, H, R) \Vdash \langle p;\ q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Applying Rule 23, we get $(C, J, L, H, R) \Vdash \langle b \to (p;\ q), \sigma\rangle \overset{\xi}{\rightsquigarrow}$.
(2) Rule 24 has been applied. Then, $\xi = \sigma \cup \xi^{\dot{C}L}$ for some $\sigma \cup \xi^{\dot{C}L}$ and $\sigma \cup \xi^{\dot{C}L} \models \neg b$. Using Rule 24 we get $(C, J, L, H, R) \Vdash \langle b \to (p;\ q), \sigma\rangle \overset{\sigma\cup\xi^{\dot{C}L}}{\rightsquigarrow}$.

Lemma 29 (Commutativity of parallel comp.) *For arbitrary closed process terms p and q we have*

$$p \parallel q \leftrightarrows q \parallel p.$$

PROOF. Let $R = \{(p \parallel q, q \parallel p) \mid p, q \in P\} \cup \{(i_d, i_d) \mid i_d \in P\}$. Since the deduction rules for $\parallel$ are symmetrical w.r.t. the left and right argument, obviously all conditions are met.

Lemma 30 (Associativity of parallel composition) *For arbitrary closed process terms p, q and r we have*

$$(p \parallel q) \parallel r \leftrightarrows p \parallel (q \parallel r).$$

PROOF. Let $R = \{((p \parallel q) \parallel r, p \parallel (q \parallel r)) \mid p, q, r \in P\} \cup \{(i_d, i_d) \mid i_d \in P\}$. The proof of the left implication of condition 1 is similar to the proof of the right implication of condition 1. The proof of condition 3 is similar to the proof of condition 2. The proofs of conditions 4 – 6 are the same as the proofs of conditions 4 – 6 of Lemma 20 (apart from the operator that has been used), because the deduction rules defined for the time transitions and the consistency predicates for $[\!]$ and $\parallel$ are the same. To increase the readability of this proof, we often apply Lemma 4 to obtain $(C, J \cup W, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ from $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ or the other way around without mentioning explicitly the use of the Lemma 4.

Condition 1: We assume $(C, J, L, H, R) \Vdash \langle (p \parallel q) \parallel r, \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle \checkmark, \sigma'\rangle$ for some C, J, L, H, R, σ, ξ, a, ξ', σ', which means that either Rule 28.1.l or Rule 28.1.r has been applied necessarily. Hence, we distinguish two cases:

(1) Rule 28.1.l has been applied. Then, we have $(C, J \cup W, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi, \mathrm{isa}(h, cs), \xi'} \langle \checkmark, \sigma\rangle$, $(C, J, L, H, R) \Vdash \langle r, \sigma\rangle \xrightarrow{\xi, \mathrm{ira}(h, cs, W), \xi'} \langle \checkmark, \sigma'\rangle$ for some W, h ,cs and $a = ca(h, cs)$. Since we do not have a rule for $(C, J \cup W, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi, \mathrm{isa}(h, cs), \xi'} \langle \checkmark, \sigma'\rangle$, we obtain a contradiction and the right implication of condition 1 holds trivially.

(2) Rule 28.1.r has been applied. Then, $(C, J \cup W, L, H, R) \Vdash \langle r, \sigma\rangle \xrightarrow{\xi, \mathrm{isa}(h, cs), \xi'} \langle \checkmark, \sigma'\rangle$, $(C, J, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi, \mathrm{ira}(h, cs, W), \xi'} \langle \checkmark, \sigma'\rangle$ for some W, h, cs, and $a = ca(h, cs)$. Since we do not have a rule for $(C, J, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi, \mathrm{ira}(h, cs, W), \xi'} \langle \checkmark, \sigma'\rangle$, we obtain a contradiction and the right implication of condition 1 holds trivially.

Condition 2: We assume $(C, J, L, H, R) \Vdash \langle (p \parallel q) \parallel r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ for some C, J, L, H, R, σ, ξ, a, ξ', k_1, σ'. Based on the deduction rule that has been applied we can distinguish ten cases:

(1) Rule 28.2.l has been applied. Then, we have $(C, J \cup W, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle k_1, \sigma'\rangle$ and $(C, J, L, H, R) \Vdash \langle r, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle \checkmark, \sigma'\rangle$ for some W, h, cs, and $a = ca(h, cs)$. For $(C, J \cup W, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle k_1, \sigma'\rangle$ we can distinguish four more cases:
   (a) Rule 29.1.l has been applied. Then, $(C, J \cup W, L, H, R) \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J \cup W, L, HR) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle \checkmark, \sigma\rangle$, $(C, J \cup W, L, H, R) \Vdash \langle q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$, and $k_1 \equiv q$. Using Rule 29.1.r we obtain $(C, J, L, H, R) \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle q, \sigma'\rangle$. Using Rule 28.3.l we obtain $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,ca(h,cs),\xi'} \langle q, \sigma'\rangle$, and observe that $(q, q) \in R$.
   (b) Rule 29.1.r has been applied. Then, $(C, J \cup W, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J \cup W, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle \checkmark, \sigma'\rangle$, $(C, J \cup W, L, H, R) \Vdash \langle p, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$, and $k_1 \equiv p$. Using Rule 28.1.l we obtain $(C, J, L, H, R) \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,ca(h,cs),\xi'} \langle \checkmark, \sigma'\rangle$. Using Rule 29.1.r we obtain $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,ca(h,cs),\xi'} \langle p, \sigma'\rangle$ and observe that $(p, p) \in R$.
   (c) Rule 29.2.l has been applied. Then, $(C, J \cup W, L, H, R) \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J \cup W, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle k_p, \sigma'\rangle$ for some k_p such that $k_1 \equiv k_p \parallel q$, and $(C, J \cup W, L, H, R) \Vdash \langle q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. Using Rule 29.1.r we obtain $(C, J, L, H, R) \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle q, \sigma'\rangle$. Using Rule 28.4.l we obtain $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,ca(h,cs),\xi'} \langle k_p \parallel q, \sigma'\rangle$. Take $k_2 \equiv k_p \parallel q$ and observe that $(k_1, k_2) \in R$.
   (d) Rule 29.2.r has been applied. Then, $(C, J \cup W, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J \cup W, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle k_q, \sigma'\rangle$ for some k_q such that $k_1 \equiv p \parallel k_q$, and $(C, J \cup W, L, H, R) \Vdash \langle p, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. Using Rule 28.2.l we obtain $(C, J, L, H, R) \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,ca(h,cs),\xi'} \langle k_q, \sigma'\rangle$. Using Rule 29.2.r we obtain $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,ca(h,cs),\xi'} \langle p \parallel k_q, \sigma'\rangle$. Take $k_2 \equiv p \parallel k_q$ and observe that $(k_1, k_2) \in R$.

(2) Rule 28.2.r has been applied. Then, $(C, J, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle \checkmark, \sigma'\rangle$ and $(C, J \cup W, L, H, R) \Vdash \langle r, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle k_1, \sigma'\rangle$ for some W, h, cs, and $a = ca(h, cs)$. This case cannot occur since the conclusion $(C, J, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle \checkmark, \sigma'\rangle$ cannot be obtained from the deduction rules.

(3) Rule 28.3.l has been applied. Then, we have $(C, J \cup W, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle \checkmark, \sigma'\rangle$ and $(C, J, L, H, R) \Vdash \langle r, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle k_1, \sigma'\rangle$

for some W, h, cs, and $a = ca(h, cs)$. The conclusion $(C, J \cup W, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle \checkmark, \sigma'\rangle$ cannot be obtained from the deduction rules. Hence, this case cannot occur.

(4) Rule 28.3.r has been applied. Then, we have $(C, J, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle k_1, \sigma'\rangle$ and $(C, J \cup W, L, H, R) \Vdash \langle r, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle \checkmark, \sigma'\rangle$ for some W, h, cs, and $a = ca(h, cs)$. For $(C, J, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle k_1, \sigma'\rangle$ we can distinguish four more cases:

(a) Rule 29.1.l has been applied. Then, we have $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle \checkmark, \sigma'\rangle$, $(C, J, L, H, R) \Vdash \langle q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$ and $k_1 \equiv q$. Using Rule 29.1.r we obtain $(C, J, L, H, R) \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle q, \sigma'\rangle$. Using Rule 28.2.r we obtain $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,ca(h,cs),\xi'} \langle q, \sigma'\rangle$ and observe that $(q, q) \in R$.

(b) Rule 29.1.r has been applied. Then, we have $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle \checkmark, \sigma'\rangle$, $(C, J, L, H, R) \Vdash \langle p, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$, and $k_1 \equiv p$. Using Rule 28.1.r we obtain $(C, J, L, H, R) \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,ca(h,cs),\xi'} \langle \checkmark, \sigma'\rangle$. Using Rule 29.1.r we obtain $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,ca(h,cs),\xi'} \langle p, \sigma'\rangle$ and observe that $(p, p) \in R$.

(c) Rule 29.2.l has been applied. Then, we have $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle k_p, \sigma'\rangle$ for some k_p such that $k_1 \equiv k_p \parallel q$, and $(C, J, L, H, R) \Vdash \langle q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. Using Rule 29.1.r we obtain $(C, J \cup W, L, H, R) \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle q, \sigma'\rangle$. Using Rule 28.4.r we obtain $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,ca(h,cs),\xi'} \langle k_p \parallel q, \sigma'\rangle$. Take $k_2 \equiv k_p \parallel q$ and observe that $(k_1, k_2) \in R$.

(d) Rule 29.2.r has been applied. Then, we have $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle k_q, \sigma'\rangle$ for some k_q such that $k_1 \equiv p \parallel k_q$, and $(C, J, L, H, R) \Vdash \langle p, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. Using Rule 28.3.r we obtain $(C, J, L, H, R) \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,ca(h,cs),\xi'} \langle k_q, \sigma'\rangle$. Using Rule 29.2.r we obtain $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,ca(h,cs),\xi'} \langle p \parallel k_q, \sigma'\rangle$. Take $k_2 \equiv p \parallel k_q$ and observe that $(k_1, k_2) \in R$.

(5) Rule 28.4.l has been applied. Then, we have $(C, J \cup W, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle k_{pq}, \sigma'\rangle$, $(C, J, L, H, R) \Vdash \langle r, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle k_r, \sigma'\rangle$ for some W, h, cs, k_{pq}, k_r such that $k_1 \equiv k_{pq} \parallel k_r$, and $a = ca(h, cs)$. For $(C, J \cup W, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle k_{pq}, \sigma'\rangle$ four cases can be distinguished:

(a) Rule 29.1.l has been applied. Then, $(C, J \cup W, L, H, R) \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J \cup W, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle \checkmark, \sigma'\rangle$, $(C, J \cup W, L, H, R) \Vdash \langle q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$, and $k_{pq} \equiv q$. Using Rule 29.2.r we obtain $(C, J, L, H, R) \Vdash$

$\langle q \parallel r, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle q \parallel k_r, \sigma'\rangle$. Using Rule 28.3.l, we obtain $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,ca(h,cs),\xi'} \langle q \parallel k_r, \sigma'\rangle$. Notice that $k_1 \equiv q \parallel k_r$. Take $k_2 \equiv q \parallel k_r$ and observe that $(k_1, k_2) \in R$.

(b) Rule 29.1.r has been applied. Then, $(C, J \cup W, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J \cup W, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle \checkmark, \sigma'\rangle$, $(C, J \cup W, L, H, R) \Vdash \langle p, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$, and $k_{pq} \equiv p$. Using Rule 28.3.l we obtain $(C, J, L, H, R) \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,\mathrm{ca}(h,cs),\xi'} \langle k_r, \sigma'\rangle$. Using Rule 29.2.r we obtain $(C,\ J,\ L,\ H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,\mathrm{ca}(h,cs),\xi'} \langle p \parallel k_r, \sigma'\rangle$. Notice that $k_1 \equiv p \parallel k_r$. Take $k_2 \equiv p \parallel k_r$ and observe that $(k_1, k_2) \in R$.

(c) Rule 29.2.l has been applied. Then, $(C, J \cup W, L, H, R) \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J \cup W, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle k_p, \sigma'\rangle$ for some k_p such that $k_{pq} \equiv k_p \parallel q$, $(C, J \cup W, L, H, R) \Vdash \langle q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. Using Rule 29.2.r we obtain $(C, J, L, H, R) \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle q \parallel k_r, \sigma'\rangle$. Using Rule 28.4.l we obtain $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,\mathrm{ca}(h,cs),\xi'} \langle k_p \parallel (q \parallel k_r), \sigma'\rangle$. Notice that $k_1 \equiv (k_p \parallel q) \parallel k_r$. Take $k_2 \equiv k_p \parallel (q \parallel k_r)$ and observe that $(k_1, k_2) \in R$.

(d) Rule 29.2.r has been applied. Then, $(C, J \cup W, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J \cup W, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle k_q, \sigma'\rangle$ for some k_q such that $k_{pq} \equiv p \parallel k_q$, and $(C, J \cup W, L, H, R) \Vdash \langle p, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. Using Rule 28.4.l we obtain $(C, J, L, H, R) \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,\mathrm{ca}(h,cs),\xi'} \langle k_q \parallel k_r, \sigma'\rangle$. Using Rule 29.2.r we obtain $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,\mathrm{ca}(h,cs),\xi'} \langle p \parallel (k_q \parallel k_r), \sigma'\rangle$. Notice that $k_1 \equiv (p \parallel k_q) \parallel k_r$. Take $k_2 \equiv p \parallel (k_q \parallel k_r)$ and observe that $(k_1, k_2) \in R$.

(6) Rule 28.4.r has been applied. Then, $(C, J, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle k_{pq}, \sigma'\rangle$, $(C, J \cup W, L, H, R) \Vdash \langle r, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle k_r, \sigma'\rangle$ for some W, h, cs, k_{pq}, k_r such that $k_1 \equiv k_{pq} \parallel k_r$, and $a = ca(h, cs)$. For $(C, J, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle k_{pq}, \sigma'\rangle$ we can distinguish four more cases:

(a) Rule 29.1.l has been applied. Then, we have $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle \checkmark, \sigma'\rangle$, $(C, J, L, H, R) \Vdash \langle q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$ and $k_{pq} \equiv q$. Using Rule 29.2.r we obtain $(C,\ J \cup W,\ L,\ H, R) \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle q \parallel k_r, \sigma'\rangle$. Using Rule 28.2.r we obtain $(C,\ J,\ L,\ H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,\mathrm{ca}(h,cs),\xi'} \langle q \parallel k_r, \sigma'\rangle$. Notice that $k_1 \equiv q \parallel k_r$. Take $k_2 \equiv q \parallel k_r$ and observe that $(k_1, k_2) \in R$.

(b) Rule 29.1.r has been applied. Then, we have $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle \checkmark, \sigma'\rangle$, $(C, J, L, H, R) \Vdash \langle p, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$ and $k_{pq} \equiv p$. Using Rule 28.2.r we obtain $(C, J, L, H, R) \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,\mathrm{ca}(h,cs),\xi'} \langle k_r, \sigma'\rangle$. Using Rule 29.2.r we obtain $(C,\ J,\ L,$

$H, R) \Vdash \langle b \to q, \sigma\rangle \overset{\xi}{\rightsquigarrow} \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,\mathrm{ca}(h,cs),\xi'} \langle p \parallel k_r, \sigma'\rangle$. Notice that $k_1 \equiv p \parallel k_r$. Take $k_2 \equiv p \parallel k_r$ and observe that $(k_1, k_2) \in R$.

(c) Rule 29.2.l has been applied. Then, we have $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle k_p, \sigma'\rangle$ for some k_p such that $k_{pq} \equiv k_p \parallel q$, and $(C, J, L, H, R) \Vdash \langle q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. Using Rule 29.2.r we obtain $(C, J \cup W, L, H, R) \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle q \parallel k_r, \sigma'\rangle$. Using Rule 28.4.r we obtain $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,\mathrm{ca}(h,cs),\xi'} \langle k_p \parallel (q \parallel k_r), \sigma'\rangle$. Notice that $k_1 \equiv (k_p \parallel q) \parallel k_r$. Take $k_2 \equiv k_p \parallel (q \parallel k_r)$ and observe that $(k_1, k_2) \in R$.

(d) Rule 29.2.r has been applied. Then, we have $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle k_q, \sigma'\rangle$ for some k_q such that $k_{pq} \equiv p \parallel k_q$, and $(C, J, L, H, R) \Vdash \langle p, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. Using Rule 28.4.r we obtain $(C,J,L, H, R) \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,\mathrm{ca}(h,cs),\xi'} \langle k_q \parallel k_r, \sigma'\rangle$. Using Rule 29.2.r we obtain $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,\mathrm{ca}(h,cs),\xi'} \langle p \parallel (k_q \parallel k_r), \sigma'\rangle$. Notice that $k_1 \equiv (p \parallel k_q) \parallel k_r$. Take $k_2 \equiv p \parallel (k_q \parallel k_r)$ and observe that $(k_1, k_2) \in R$.

(7) Rule 29.1.l has been applied. Then, $(C, J, L, H, R) \Vdash \langle r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$, $k_1 \equiv r$, and $(C, J, L, H, R) \Vdash \langle r, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. Then two cases can be considered:

(a) Rule 28.1.l has been applied. Then, we have $(C, J \cup W, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle \checkmark, \sigma'\rangle$, $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle \checkmark, \sigma'\rangle$ for some W, h, cs and $a = ca(h, cs)$. Using Rule 29.1.l we obtain $(C, J, L, H, R) \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle r, \sigma'\rangle$. Using Rule 28.3.l we obtain $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,ca(h,cs),\xi'} \langle r, \sigma'\rangle$, and observe that $(r, r) \in R$.

(b) Rule 28.1.r has been applied. Then, we have $(C, J \cup W, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle \checkmark, \sigma'\rangle$, $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,\mathrm{ira}(h,cs,W),\xi'} \langle \checkmark, \sigma'\rangle$ for some W, h, cs, and $a = ca(h, cs)$. Then, using Rule 29.1.l we obtain $(C, J \cup W, L, H, R)$ $\langle q \parallel r, \sigma\rangle \xrightarrow{\xi,\mathrm{isa}(h,cs),\xi'} \langle r, \sigma'\rangle$. Using Rule 28.3.r we obtain $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,ca(h,cs),\xi'} \langle r, \sigma'\rangle$, and observe that $(r, r) \in R$.

(8) Rule 29.1.r has been applied. Then, we have $(C, J, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$, $(C, J, L, H, R) \Vdash \langle p \parallel q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$, and $k_1 \equiv p \parallel q$. According to Rule 31, we have $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle p, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$, and $(C, J, L, H, R) \Vdash \langle q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. Using Rule 29.1.r, we obtain $(C, J, L, H, R) \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle q, \sigma'\rangle$. Then, using Rule 29.2.r, we obtain $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle p \parallel q, \sigma'\rangle$. Take $k_2 \equiv p \parallel q$ and observe that $(k_1, k_2) \in R$.

(9) Rule 29.2.l has been applied. Then, $(C,\, J,\, L,\, H, R) \Vdash \langle r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C,\, J,\, L,\, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle k_{pq}, \sigma'\rangle$ for some term k_{pq} such that $k_1 \equiv k_{pq} \parallel r$ and $(C, J, L, H, R) \Vdash \langle r, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. For $(C, J, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle k_{pq}, \sigma'\rangle$, ten cases can be distinguished.

(a) Rule 28.2.l has been applied. Then, we have $(C, J, L, H, R) \Vdash \langle r, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J \cup W, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi, \mathrm{isa}(h,cs), \xi'} \langle k_{pq}, \sigma'\rangle$ and $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi, \mathrm{ira}(h,cs,W), \xi'} \langle \checkmark, \sigma'\rangle$ for some W, h, cs, and $a = ca(h, cs)$. Then applying Rule 29.1.l followed by Rule 28.4.l gives $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi, ca(h,cs), \xi'} \langle k_{pq} \parallel r, \sigma'\rangle$. Take $k_2 \equiv k_{pq} \parallel r$ and observe that $(k_1, k_2) \in R$.

(b) Rule 28.2.r has been applied. Then, we have $(C, J \cup W, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi, \mathrm{isa}(h,cs), \xi'} \langle \checkmark, \sigma'\rangle$ and $(C,\, J,\, L,\, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi, \mathrm{ira}(h,cs,W), \xi'} \langle k_{pq}, \sigma'\rangle$ for some W, h, cs, and $a = ca(h,cs)$. Then Rule 29.1.l followed by Rule 28.4.r gives $(C,\, J,\, L,\, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi, ca(h,cs), \xi'} \langle k_{pq} \parallel r, \sigma'\rangle$. Take $k_2 \equiv k_{pq} \parallel r$ and observe that $(k_1, k_2) \in R$.

(c) Rule 28.3.l has been applied. Then, we have $(C, J \cup W, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi, \mathrm{isa}(h,cs), \xi'} \langle \checkmark, \sigma'\rangle$ and $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi, \mathrm{ira}(h,cs,W), \xi'} \langle k_{pq}, \sigma'\rangle$ for some W, h, cs, and $a = ca(h, cs)$. Then Rule 29.2.l followed by Rule 28.3.l gives $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi, ca(h,cs), \xi'} \langle k_{pq} \parallel r, \sigma'\rangle$. Take $k_2 \equiv k_{pq} \parallel r$ and observe that $(k_1, k_2) \in R$.

(d) Rule 28.3.r has been applied. Then, we have $(C, J \cup W, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi, \mathrm{isa}(h,cs), \xi'} \langle k_{pq}, \sigma'\rangle$ and $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi, \mathrm{ira}(h,cs,W), \xi'} \langle \checkmark, \sigma'\rangle$ for some W, h, cs, $a = ca(h, cs)$. Then applying Rule 29.1.l followed by Rule 28.4.r gives $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi, ca(h,cs), \xi'} \langle k_{pq} \parallel r, \sigma'\rangle$. Take $k_2 \equiv k_{pq} \parallel r$ and observe that $(k_1, k_2) \in R$.

(e) Rule 28.4.l has been applied. Then, we have $(C, J \cup W, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi, \mathrm{isa}(h,cs), \xi'} \langle k_p, \sigma'\rangle$, $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi, \mathrm{ira}(h,cs,W), \xi'} \langle k_q, \sigma'\rangle$ for some W, h, cs, k_p, k_q such that $k_{pq} \equiv k_p \parallel k_q$, and $a = ca(h, cs)$. Then applying Rule 29.2.l followed by Rule 28.4.l gives $(C,\, J,\, L,\, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi, ca(h,cs), \xi'} \langle k_p \parallel (k_q \parallel r), \sigma'\rangle$. Notice that $k_1 \equiv (k_p \parallel k_q) \parallel r$. Take $k_2 \equiv k_p \parallel (k_q \parallel r)$ and observe that $(k_1, k_2) \in R$.

(f) Rule 28.4.r has been applied. Then, we have $(C, J \cup W, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi, \mathrm{isa}(h,cs), \xi'} \langle k_q, \sigma'\rangle$ and $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi, \mathrm{ira}(h,cs,W), \xi'} \langle k_p, \sigma'\rangle$ for some W, h, cs, k_p, k_q such that $k_{pq} \equiv k_p \parallel k_q$, and $a = ca(h, cs)$. Then applying Rule 29.2.l followed by Rule 28.4.r gives $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle k_p \parallel (k_q \parallel r), \sigma'\rangle$. Notice that $k_1 \equiv (k_p \parallel k_q) \parallel r$. Take $k_2 \equiv k_p \parallel (k_q \parallel r)$ and observe that $(k_1, k_2) \in R$.

(g) Rule 29.1.l has been applied. Then, we have $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi, a, \xi'} \langle \checkmark, \sigma'\rangle$, $(C, J, L, H, R) \Vdash \langle q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$,

and $k_{pq} \equiv q$. We have $(C,J, L,H,R) \Vdash \langle p \parallel q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and $(C, J, L, H, R) \Vdash \langle p \parallel q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$ (see Rule 31). Applying Rule 29.1.l gives $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle q \parallel r, \sigma'\rangle$. Notice that $k_1 \equiv q \parallel r$. Take $k_2 \equiv q \parallel r$ and observe that $(k_1, k_2) \in R$.

(h) Rule 29.1.r has been applied. Then, we have $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$, $(C, J, L, H, R) \Vdash \langle p, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$ and $k_{pq} \equiv p$. Applying Rule 29.1.l and then Rule 29.2.r gives $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle p \parallel r, \sigma'\rangle$. Notice that $k_1 \equiv q \parallel r$. Take $k_2 \equiv p \parallel r$ and observe that $(k_1, k_2) \in R$.

(i) Rule 29.2.l has been applied. Then, we have $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_p, \sigma'\rangle$ for some k_p such that $k_{pq} \equiv k_p \parallel q$, and $(C, J, L, H, R) \Vdash \langle q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. We have $(C, J, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ and $(C, J, L, H, R) \Vdash \langle p \parallel q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$ (see Rule 31). Applying Rule 29.2.l gives $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_p \parallel (q \parallel r), \sigma'\rangle$. Take $k_2 \equiv k_p \parallel (q \parallel r)$ and observe that $(k_1, k_2) \in R$.

(j) Rule 29.2.r has been applied. Then, we have $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_q, \sigma'\rangle$ for some k_q such that $k_{pq} \equiv p \parallel k_q$, and $(C, J, L, H, R) \Vdash \langle p, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. Applying Rule 29.2.l and then Rule 29.2.r gives (C, J, L, H, R) $\langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle p \parallel (k_q \parallel r), \sigma'\rangle$. Notice $k_1 \equiv (p \parallel k_q) \parallel r$. Take $k_2 \equiv p \parallel (k_q \parallel r)$ and observe that $(k_1, k_2) \in R$.

(10) Rule 29.2.r has been applied. Then, we have $(C, J, L, H, R) \Vdash \langle p \parallel q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_r, \sigma'\rangle$ for some k_r such that $k_1 \equiv (p \parallel q) \parallel k_r$, and $(C, J, L, H, R) \Vdash \langle p \parallel q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. According to Rule 31, we have $(C, J, L, H, R) \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle q, \sigma\rangle \overset{\xi}{\rightsquigarrow}$, $(C, J, L, H, R) \Vdash \langle p, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$ and $(C, J, L, H, R) \Vdash \langle q, \sigma'\rangle \overset{\xi'}{\rightsquigarrow}$. Using Rule 29.2.r, we obtain $(C, J, L, H, R) \Vdash \langle q \parallel r, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle q \parallel k_r, \sigma'\rangle$. Using Rule 29.2.r, we obtain $(C, J, L, H, R) \Vdash \langle p \parallel (q \parallel r), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle p \parallel (q \parallel k_r), \sigma'\rangle$. Take $k_2 \equiv p \parallel (q \parallel k_r)$ and observe that $(k_1, k_2) \in R$.

B.6 Properties of action encapsulation operator

Lemma 31 *For arbitrary closed process term p we have*

$$\partial_\emptyset(p) \leftrightarrows p.$$

PROOF. Let $R = \{(\partial_\emptyset(p), p) \mid p \in P\} \cup \{(i_d, i_d) \mid i_d \in P\}$.

Condition 1: First, we assume $E \Vdash \langle \partial_\emptyset(p), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ for some E, σ, ξ, a, ξ', σ', which means Rule 32.1 has been applied necessarily. Then, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$. Second, we assume $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ for some $E, \sigma, \xi, a, \xi', \sigma'$. We know that $a \notin \emptyset$. Using Rule 32.1, we obtain $E \Vdash \langle \partial_\emptyset(p), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$.

Condition 2: We assume $E \Vdash \langle \partial_\emptyset(p), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ for some $E, \sigma, \xi, a, \xi', k_1, \sigma'$, which means Rule 32.2 has been applied necessarily. Then, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_p, \sigma'\rangle$ for some k_p such that $k_1 \equiv \partial_\emptyset(k_p)$. Take $k_2 \equiv k_p$ and observe that $(k_1, k_2) \in R$.

Condition 3: We assume $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ for some $E, \sigma, \xi, a, \xi', k_1, \sigma'$. We know that $a \notin \emptyset$. Using Rule 32.2, we obtain $E \Vdash \langle \partial_\emptyset(p), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \partial_\emptyset(k_1), \sigma'\rangle$. Take $k_2 \equiv \partial_\emptyset(k_1)$ and observe that $(k_2, k_1) \in R$.

Condition 4: We assume $E \Vdash \langle \partial_\emptyset(p), \sigma\rangle \xmapsto{t,\rho} \langle k_1, \sigma'\rangle$ for some $E, \sigma, t, \rho, k_1, \sigma'$, which means Rule 33 has been applied necessarily. Then, $E \Vdash \langle p, \sigma\rangle \xmapsto{t,\rho} \langle k_p, \sigma'\rangle$ for some k_p such that $k_1 \equiv \partial_\emptyset(k_p)$. Take $k_2 \equiv k_p$ and observe that $(k_1, k_2) \in R$.

Condition 5: We assume $E \Vdash \langle p, \sigma\rangle \xmapsto{t,\rho} \langle k_1, \sigma'\rangle$ for some $E, \sigma, t, \rho, k_1, \sigma'$. Using Rule 33, we obtain $E \Vdash \langle \partial_\emptyset(p), \sigma\rangle \xmapsto{t,\rho} \langle \partial_\emptyset(k_1), \sigma'\rangle$. Take $k_2 \equiv \partial_\emptyset(k_1)$ and observe that $(k_2, k_1) \in R$.

Condition 6: First, we assume $E \Vdash \langle \partial_\emptyset(p), \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some E, σ, ξ, which means Rule 34 has been applied necessarily. Then, $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Second, we assume $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some E, σ, ξ. Using Rule 34, we obtain $E \Vdash \langle \partial_\emptyset(p), \sigma\rangle \overset{\xi}{\rightsquigarrow}$.

Lemma 32 *For arbitrary closed process term p and sets of actions A and A' we have*

$$\partial_A(\partial_{A'}(p)) \leftrightarrows \partial_{A\cup A'}(p).$$

PROOF. Let $R = \{(\partial_A(\partial_{A'}(p)), \partial_{A\cup A'}(p)) \mid p \in P, \text{sets of actions } A, A'\} \cup \{(i_d, i_d) \mid i_d \in P\}$.

Condition 1: First, we assume $E \Vdash \langle \partial_A(\partial_{A'}(p)), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ for some $E, \sigma, \xi, a, \xi', \sigma'$, which means Rule 32.1 has been applied necessarily. Then, $E \Vdash \langle \partial_{A'}(p), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ and $a \notin A$. Again, due to Rule 32.1, we have $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ and $a \notin A'$. From $a \notin A$ and $a \notin A'$, we know that $a \notin A \cup A'$. Using Rule 32.1, we obtain $E \Vdash \langle \partial_{A\cup A'}(p), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$. Second, we assume $E \Vdash \langle \partial_{A\cup A'}(p), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \surd, \sigma'\rangle$ for some $E, \sigma, \xi, a, \xi', \sigma'$, which

means Rule 32.1 has been applied necessarily. Then, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$ and $a \notin A \cup A'$. From $a \notin A \cup A'$, we know that $a \notin A$ and $a \notin A'$. Using Rule 32.1, we get $E \Vdash \langle \partial_{A'}(p), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$. Again, using Rule 32.1, we obtain $E \Vdash \langle \partial_A(\partial_{A'}(p)), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \checkmark, \sigma'\rangle$.

Condition 2: We assume $E \Vdash \langle \partial_A(\partial_{A'}(p)), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ for some E, σ, ξ, a, ξ', k_1, σ', which means Rule 32.2 has been applied necessarily. Then, $E \Vdash \langle \partial_{A'}(p), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_p, \sigma'\rangle$ for some k_p such that $k_1 \equiv \partial_A(k_p)$ and $a \notin A$. Using Rule 32.2, we get $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k'_p, \sigma'\rangle$ for some k'_p such that $k_p \equiv \partial_{A'}(k'_p)$ and $a \notin A'$. From $a \notin A$ and $a \notin A'$, we know that $a \notin A \cup A'$. Using Rule 32.2, we get $E \Vdash \langle \partial_{A\cup A'}(p), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \partial_{A\cup A'}(k'_p), \sigma'\rangle$. Note that $k_1 \equiv \partial_A(\partial_{A'}(k'_p))$. Take $k_2 \equiv \partial_{A\cup A'}(k'_p)$ and observe that $(k_1, k_2) \in R$.

Condition 3: We assume $E \Vdash \langle \partial_{A\cup A'}(p), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_1, \sigma'\rangle$ for some E, σ, ξ, a, ξ', k_1, σ', which means Rule 32.2 has been applied necessarily. Then, $E \Vdash \langle p, \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle k_p, \sigma'\rangle$ for some k_p such that $k_1 \equiv \partial_{A\cup A'}(k_p)$ and $a \notin A \cup A'$. From $a \notin A \cup A'$, we know that $a \notin A$ and $a \notin A'$. Using Rule 32.2, we get $E \Vdash \langle \partial_{A'}(p), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \partial_{A'}(k_p), \sigma'\rangle$. Again, due to Rule 32.2, we have $E \Vdash \langle \partial_A(\partial_{A'}(p)), \sigma\rangle \xrightarrow{\xi,a,\xi'} \langle \partial_A(\partial_{A'}(k_p)), \sigma'\rangle$. Take $k_2 \equiv \partial_A(\partial_{A'}(k_p))$ and observe that $(k_2, k_1) \in R$.

Condition 4: We assume $E \Vdash \langle \partial_A(\partial_{A'}(p)), \sigma\rangle \xmapsto{t,\rho} \langle k_1, \sigma'\rangle$ for some E, σ, t, ρ, k_1, σ', which means Rule 33 has been applied necessarily. Then, $E \Vdash \langle \partial_{A'}(p), \sigma\rangle \xmapsto{t,\rho} \langle k_p, \sigma'\rangle$ for some k_p such that $k_1 \equiv \partial_A(k_p)$. Again, due to Rule 33, we get $E \Vdash \langle p, \sigma\rangle \xmapsto{t,\rho} \langle k'_p, \sigma'\rangle$ for some k'_p such that $k_p \equiv \partial_{A'}(k'_p)$. Using Rule 33, we obtain $E \Vdash \langle \partial_{A\cup A'}(p), \sigma\rangle \xmapsto{t,\rho} \langle \partial_{A\cup A'}(k'_p), \sigma'\rangle$. Note that $k_1 \equiv \partial_A(\partial_{A'}(k'_p))$. Take $k_2 \equiv \partial_{A\cup A'}(k'_p)$ and observe that $(k_1, k_2) \in R$.

Condition 5: We assume $E \Vdash \langle \partial_{A\cup A'}(p), \sigma\rangle \xmapsto{t,\rho} \langle k_1, \sigma'\rangle$ for some $E, \sigma, t, \rho, k_1, \sigma'$, which means Rule 33 has been applied necessarily. Then, $E \Vdash \langle p, \sigma\rangle \xmapsto{t,\rho} \langle k_p, \sigma'\rangle$ for some k_p such that $k_1 \equiv \partial_{A\cup A'}(k_p)$. Using Rule 33, we get $E \Vdash \langle \partial_{A'}(p), \sigma\rangle \xmapsto{t,\rho} \langle \partial_{A'}(k_p), \sigma'\rangle$. Due to Rule 33, we obtain $E \Vdash \langle \partial_A(\partial_{A'}(p)), \sigma\rangle \xmapsto{t,\rho} \langle \partial_A(\partial_{A'}(k_p)), \sigma'\rangle$. Take $k_2 \equiv \partial_A(\partial_{A'}(k_p))$ and observe that $(k_2, k_1) \in R$.

Condition 6: First, we assume $E \Vdash \langle \partial_A(\partial_{A'}(p)), \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some E, σ, ξ, which means Rule 34 has been applied necessarily. Then, $E \Vdash \langle \partial_{A'}(p), \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Again, due to Rule 34, we have $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Then using Rule 34, we obtain $E \Vdash \langle \partial_{A\cup A'}(p), \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Second, we assume $E \Vdash \langle \partial_{A\cup A'}(p), \sigma\rangle \overset{\xi}{\rightsquigarrow}$ for some E, σ, ξ. which means Rule 34 has been applied necessarily. Then, $E \Vdash \langle p, \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 34, we obtain $\langle E \Vdash \partial_{A'}(p), \sigma\rangle \overset{\xi}{\rightsquigarrow}$. Using Rule 34 again, we obtain $E \Vdash$

$\langle \partial_A(\partial_{A'}(p)), \sigma \rangle \overset{\xi}{\rightsquigarrow}$.

B.7 Inconsistent process

Lemma 33 *For arbitrary predicate u, using the previous properties gives,*

$$u \curvearrowright \bot \underline{\leftrightarrow} \bot.$$

PROOF. $u \curvearrowright \bot \leftrightarrow u \curvearrowright (\text{false} \curvearrowright p) \underline{\leftrightarrow} (u \wedge \text{false}) \curvearrowright p \underline{\leftrightarrow} \text{false} \curvearrowright p \underline{\leftrightarrow} \bot$.

Lemma 34 *For arbitrary closed term p we have*

$$p \between \bot \underline{\leftrightarrow} \bot.$$

PROOF. Since there are no transition rules defined for $\bot$, also note that $p \between \bot$ has no transitions, the conditions 1 – 6 hold trivially.

Lemma 35 *For arbitrary closed process term p we have*

$$p \parallel \bot \underline{\leftrightarrow} \bot.$$

PROOF. Since there are no transition rules defined for $\bot$, also note that $p \parallel \bot$ has no transitions, the conditions 1 – 6 hold trivially.

Lemma 36 *For arbitrary set of actions A we have*

$$\partial_A(\bot) \underline{\leftrightarrow} \bot.$$

PROOF. Since there are no transition rules defined for $\bot$, also note that $\partial_A(\bot)$ has no transitions, the conditions 1 – 6 hold trivially.

Lemma 37 *For arbitrary closed process term p we have*

$$\bot; p \underline{\leftrightarrow} \bot.$$

PROOF. Since there are no transition rules defined for $\bot$, also note that $p; \bot$ has no transitions, the conditions 1 – 6 hold trivially.

Lemma 38 *We have*

$$\text{skip}; \bot \underline{\leftrightarrow} \delta.$$

PROOF. We know that skip $\equiv \emptyset : \text{true} \gg \tau$. Let $R = \{(\emptyset : \text{true} \gg \tau;\ \bot, \delta)\}$. Since there are no action transition rules and time transition rules defined for δ and $\bot$, also $\emptyset : \text{true} \gg \tau;\ \bot$ cannot perform any action transitions (because $\bot$ is not consistent) and time transitions (because no time transition rules defined for $\emptyset : \text{true} \gg \tau$), the conditions 1 – 5 hold trivially.

Condition 6: First, we assume $(C, J, L, H, R) \Vdash \langle \emptyset : \text{true} \gg \tau;\ \bot, \sigma \rangle \overset{\xi}{\rightsquigarrow}$ for some $C, J, L, H, R, \sigma, \xi$, which means that Rule 19 has been applied necessarily. Then, $(C, J, L, H, R) \Vdash \langle \emptyset : \text{true} \gg \tau, \sigma \rangle \overset{\xi}{\rightsquigarrow}$ such that $\xi = \sigma \cup \xi^{\dot{C}L}$ for some $\sigma \cup \xi^{\dot{C}L}$ (see also Rule 2). Then, we get $(C, J, L, H, R) \Vdash \langle \delta, \sigma \rangle \overset{\sigma \cup \xi^{\dot{C}L}}{\rightsquigarrow}$ using Rule 9. Second, we assume $(C, J, L, H, R) \Vdash \langle \delta, \sigma \rangle \overset{\xi}{\rightsquigarrow}$, which means that Rule 9 has been applied necessarily. Then $\xi = \sigma \cup \xi^{\dot{C}L}$ for some $\sigma \cup \xi^{\dot{C}L}$. Using Rule 2, we get $(C, J, L, H, R) \Vdash \langle \emptyset : \text{true} \gg \tau, \sigma \rangle \overset{\sigma \cup \xi^{\dot{C}L}}{\rightsquigarrow}$. According to Rule 19, we obtain $(C, J, L, H, R) \Vdash \langle \emptyset : \text{true} \gg \tau;\ \bot, \sigma \rangle \overset{\sigma \cup \xi^{\dot{C}L}}{\rightsquigarrow}$.

Lemma 39 *Using the previous properties gives,*

$$\bot \underline{\leftrightarrow} \text{ false}.$$

PROOF. $\bot \underline{\leftrightarrow} \text{ false} \curvearrowright \text{false} \underline{\leftrightarrow} \text{ false}$.