Negative premises in applied process calculi

Johannes Åman Pohjola, Johannes Borgström, Joachim Parrow, Palle Raabjerg, and Ioana Rodhe

Department of Information Technology, Uppsala University, Sweden

Abstract. We explore two applications of negative premises to increase the expressive power of psi-calculi: reliable broadcasts and priorities. Together, these can be used to model discrete time, which we illustrate with an example from automotive applications. The negative premises can be encoded by a two-level structural operational semantics without negative premises; we use this fact to prove the standard congruence and structural laws of bisimulation with Nominal Isabelle.

1 Introduction

A *negative premise* in a structural operational semantics (SOS) rule states that in order to derive a transition, the absence of another transition must be proven. Applications include priorities, deadlock detection and sequential composition: an event can only occur if a higher-priority event is *not* available, a deadlock is detected if another transition can *not* be derived, and $P; Q$ may proceed as Q iff P can *not* act. Process algebras with negative premises have been studied for more than 20 years. The main novel contribution of this paper is to use negative premises in a general framework of high-level applied process calculi, where it is easy to define highly specialised modelling languages for particular applications. We show how to capture phenomena such as priorities and reliable broadcasts, and demonstrate the applicability of our framework through a nontrivial example.

Two early studies of negative premises in structural operational semantics are Bloom et al. [4] and Groote [12]; van Glabbeek [28] gives a more recent exposition of the involved challenges. One obvious problem is that it is possible to give an inconsistent set of rules, where a process has a transition if and only if it does not. Groote's solution involves introducing the notion of a *stratification*. Formally, a stratification is a function S from transitions to some ordinal such that for every instance of a rule application that may occur in a derivation tree, the negative premises have smaller S-values than the conclusion, and the positive premises have no larger S-values than the conclusion. A corollary of the existence of a stratification is that there is no transition whose absence is a precondition for its presence. Transitions are constructed stratum by stratum in a bottom-up fashion, where negative premises on a given stratum are true when the corresponding positive premises are not provable on any of the lower strata.

All previous work on negative premises known to us apply only to basic process calculi where there are no high-level data structures or logics in the process syntax. Psi-calculi [3] is a parametric framework for extensions of the pi-calculus, with arbitrary data structures and logical assertions for facts about data. In earlier papers we have shown how psi-calculi can capture the same phenomena as other proposed extensions of the pi-calculus such as the applied pi-calculus, the spi-calculus, the fusion calculus, the concurrent constraint pi-calculus, and calculi with polyadic communication channels or pattern matching. Psi-calculi can be even more general, for example by allowing structured channels, higher-order formalisms such as the lambda calculus for data structures, and predicate logic for assertions.

In this paper, we extend psi-calculi with two examples of negative premises. First, we introduce a reliable broadcast communication, where the possibility of message loss is prevented by a negative premise in the rule for execution of parallel processes. In other words, it is impossible for a message to miss a process that listens for it. Second, we introduce a priority system, where negative premises ensure that synchronisation over higher priority channels block actions of lower priority. In both cases, we gain generality by making the reliability and priority of channels a dynamic property which may depend on the current process environment. The precise nature of this dependency may vary between different uses of broadcasts and priorities, subject to some natural restrictions.

For reliable broadcast and priorities, the structural operational semantics with negative premises coincides with a two-layer formulation, where the negative premises in the top layer are expressed in terms of the bottom layer, and the bottom layer only uses positive premises — and hence is a formulation that formally has no negative premises. We use this approach when formalising our framework in Nominal Isabelle, allowing us to directly reuse earlier work on the topic by Bengtson [2]. We obtain machine-checked proofs that strong bisimulation satisfies the expected algebraic properties.

We evaluate our framework by showing that a notion of discrete time emerges as a special case of low-priority reliable broadcasts of clock signals. We demonstrate that instances of psi-calculi can capture the broadcast pi-calculus [10] and model the CAN protocol for bus arbitration in automotive vehicles [9].

2 Background on Psi-calculi

The following is a quick recapitulation of the psi-calculi framework. For an in-depth introduction with motivations and examples refer the reader to [3].

We assume a countably infinite set of atomic *names* $\mathcal{N}$ ranged over by $a, b, \ldots, z$. Intuitively, names will represent the symbols that can be scoped, and also represent symbols acting as variables in the sense that they can be subject to substitution. A *nominal set* [21] is a set equipped with a formal notion of what it means for a name a to occur in an element X of the set, written $a \in \mathrm{n}(X)$ (often pronounced as "a is in the support of X"). We write $a\#X$, pronounced "a is fresh for X", for $a \notin \mathrm{n}(X)$, and if A is a set of names we write $A\#X$ to mean $\forall a \in A \,.\, a\#X$. In the following $\tilde{a}$ means a finite sequence of names, $a_1, \ldots, a_n$. The empty sequence is written ϵ and the concatenation of $\tilde{a}$ and $\tilde{b}$ is written $\tilde{a}\tilde{b}$.

When occurring as an operand of a set operator, $\tilde{a}$ means the corresponding set of names $\{a_1, \dots, a_n\}$. We also use sequences of other nominal sets in the same way. For names, we write $(\widetilde{a}\,\widetilde{b})$ for the *name swapping* that swaps each element of $\widetilde{a}$ with the corresponding element of $\widetilde{b}$; here it is implicit that $\widetilde{a}$ and $\widetilde{b}$ have the same length, and that the names in $\widetilde{a}$ (resp. $\widetilde{b}$) are pair-wise distinct.

A *nominal datatype* is a nominal set together with a set of functions on it. In particular we shall consider substitution functions that substitute elements for names. If X is an element of a datatype, $\tilde{a}$ is a sequence of names without duplicates and $\tilde{Y}$ is an equally long sequence of elements of possibly another datatype, the *substitution* $X[\tilde{a} := \tilde{Y}]$ is an element of the same datatype as X. Substitution is required to satisfy a law akin to alpha-conversion: if $\widetilde{b}\#X, \widetilde{a}$ then $X[\widetilde{a} := \widetilde{T}] = ((\widetilde{b}\,\widetilde{a}) \cdot X)[\widetilde{b} := \widetilde{T}]$.

A psi-calculus is defined by instantiating three nominal data types and four operators:

Definition 1 (Psi-calculus parameters). *A psi-calculus requires the three (not necessarily disjoint) nominal data types: the (data) terms* $\mathbf{T}$*, ranged over by* M, N*, the conditions* $\mathbf{C}$*, ranged over by* φ*, the assertions* $\mathbf{A}$*, ranged over by* Ψ*, and the four equivariant operators:*

$$\begin{array}{ll}
\dot{\leftrightarrow} \in \mathbf{T} \times \mathbf{T} \to \mathbf{C} & \textit{(Unicast) Channel Equivalence} \\
\otimes \in \mathbf{A} \times \mathbf{A} \to \mathbf{A} & \textit{Composition} \\
\mathbf{1} : \mathbf{A} & \textit{Unit} \\
\vdash \subseteq \mathbf{A} \times \mathbf{C} & \textit{Entailment}
\end{array}$$

and substitution functions $[\widetilde{a} := \widetilde{M}]$, substituting terms for names, on each of $\mathbf{T}$, $\mathbf{C}$ and $\mathbf{A}$, where the substitution function on $\mathbf{T}$, in addition to the alpha-conversion-like law above, satisfies the following name preservation laws: if $\widetilde{a} \subseteq \mathsf{n}(M)$ and $b \in \mathsf{n}(\widetilde{N})$ then $b \in \mathsf{n}(M[\widetilde{a} := \widetilde{N}])$; and if $b \in \mathsf{n}(M)$ and $b\#\widetilde{a}, \widetilde{N}$ then $b \in \mathsf{n}(M[\widetilde{a} := \widetilde{N}])$.

The binary functions above will be written in infix. Thus, $M \dot{\leftrightarrow} N$ is a condition, pronounced "M and N are channel equivalent". We write $\Psi \vdash \varphi$, "Ψ entails φ", for $(\Psi, \varphi) \in \vdash$, and if Ψ and Ψ' are assertions then so is $\Psi \otimes \Psi'$.

We say that two assertions are equivalent, written $\Psi \simeq \Psi'$ if they entail the same conditions, i.e. for all φ we have that $\Psi \vdash \varphi \Leftrightarrow \Psi' \vdash \varphi$. We impose certain requisites on the sets and operators. In brief, channel equivalence must be symmetric and transitive, $\otimes$ must be compositional with regard to $\simeq$, and the assertions with $(\otimes, \mathbf{1})$ form an abelian monoid modulo $\simeq$.

A *frame* F can intuitively be thought of as an assertion with local names: it is of the form $(\nu\widetilde{b})\Psi$ where $\widetilde{b}$ is a sequence of names that bind into the assertion Ψ. We use F, G to range over frames. We overload Ψ to also mean the frame $(\nu\epsilon)\Psi$ and $\otimes$ to composition on frames defined by $(\nu\widetilde{b_1})\Psi_1 \otimes (\nu\widetilde{b_2})\Psi_2 = (\nu\widetilde{b_1}\widetilde{b_2})(\Psi_1 \otimes \Psi_2)$ where $\widetilde{b_1}\#\widetilde{b_2}, \Psi_2$ and vice versa. We write $\Psi \otimes F$ to mean $(\nu\epsilon)\Psi \otimes F$, and $(\nu c)((\nu\widetilde{b})\Psi)$ for $(\nu c\widetilde{b})\Psi$.

Alpha equivalent frames are identified. We define $F \vdash \varphi$ to mean that there exists an alpha variant $(\nu\widetilde{b})\Psi$ of F such that $\widetilde{b}\#\varphi$ and $\Psi \vdash \varphi$. We also define $F \simeq G$ to mean that for all φ it holds that $F \vdash \varphi$ iff $G \vdash \varphi$. Intuitively a condition is entailed by a frame if it is entailed by the assertion and does not contain any names bound by the frame, and two frames are equivalent if they entail the same conditions.

Definition 2 (Psi-calculus agents). *Given psi-calculus parameters as in Definition 1, the* agents*, ranged over by $P, Q, \ldots$, are of the following forms.*

$\mathbf{0}$	Nil
$\overline{M}N\,.\,P$	Output
$\underline{M}(\lambda\widetilde{x})N\,.\,P$	Input
$\mathbf{case}\ \varphi_1 : P_1\ [\!]\ \cdots\ [\!]\ \varphi_n : P_n$	Case
$(\nu a)P$	Restriction
$P \mid Q$	Parallel
$!P$	Replication
$(\!\vert\Psi\vert\!)$	Assertion

Restriction binds a in P and Input binds $\widetilde{x}$ in both N and P. We identify alpha equivalent agents. An occurrence of a subterm in an agent is guarded *if it is a proper subterm of a Prefix form. An agent is* assertion guarded *if it contains no unguarded Assertions. An agent is* well-formed *if in $\underline{M}(\lambda\widetilde{x})N.P$ it holds that $\widetilde{x} \subseteq \mathrm{n}(N)$ is a sequence without duplicates, that in a replication $!P$ the agent P is assertion guarded, and that in* **case** *$\varphi_1 : P_1\ [\!]\ \cdots\ [\!]\ \varphi_n : P_n$ the agents P_i are assertion guarded.*

The agent **case** $\varphi_1 : P_1\ [\!]\ \cdots\ [\!]\ \varphi_n : P_n$ is sometimes abbreviated as **case** $\widetilde{\varphi} : \widetilde{P}$, or if $n = 1$ as **if** φ_1 **then** P_1. Input subjects are underlined to facilitate parsing of complicated expressions; in simple cases we often omit the underline. We sometimes write $\underline{M}(x).P$ for $\underline{M}(\lambda x)x.P$. From this point on, we only consider well-formed agents.

The *frame $\mathcal{F}(P)$ of an agent* P is defined inductively as follows:

$$\mathcal{F}(\underline{M}(\lambda\widetilde{x})N\,.\,P) = \mathcal{F}(\overline{M}\,N\,.\,P) = \mathcal{F}(\mathbf{0}) = \mathcal{F}(\mathbf{case}\ \widetilde{\varphi} : \widetilde{P}) = \mathcal{F}(!P) = \mathbf{1}$$
$$\mathcal{F}((\!\vert\Psi\vert\!)) = (\nu\epsilon)\Psi \qquad \mathcal{F}(P \mid Q) = \mathcal{F}(P) \otimes \mathcal{F}(Q) \qquad \mathcal{F}((\nu b)P) = (\nu b)\mathcal{F}(P)$$

The *actions* ranged over by α, β are of the following three kinds: *Output* $\overline{M}(\nu\tilde{a})N$ where $\tilde{a} \subseteq \mathrm{n}(N)$, *input* $\underline{M}\,N$, where $\tilde{a} \subseteq \mathrm{n}(N)$, and *silent* $\tau : p$. Here we refer to M as the *subject* and N as the *object*. We define $\mathrm{bn}(\overline{M}(\nu\tilde{a})N) = \mathrm{bn}(!\overline{M}\ (\nu\tilde{a})N) = \tilde{a}$, and $\mathrm{bn}(\alpha) = \emptyset$ if α is an input, broadcast input or $\tau : p$. We also define $\mathrm{n}(\tau : p) = \emptyset$ and $\mathrm{n}(\alpha) = \mathrm{n}(M) \cup \mathrm{n}(N)$ for the input and output actions. As in the pi-calculus, the output $\overline{M}(\nu\tilde{a})N$ represents an action sending N along M and opening the scopes of the names $\tilde{a}$. Note in particular that the support of this action includes $\tilde{a}$. Thus $\overline{M}(\nu a)a$ and $\overline{M}(\nu b)b$ are different actions.

$$\textsc{In}\ \frac{\Psi \vdash K \leftrightarrow M}{\Psi \rhd \underline{M}(\lambda\widetilde{y})N\,.\,P \xrightarrow{\underline{K}\,N[\widetilde{y}:=\widetilde{L}]} P[\widetilde{y}:=\widetilde{L}]} \qquad \textsc{Out}\ \frac{\Psi \vdash M \leftrightarrow K}{\Psi \rhd \overline{M}\,N\,.\,P \xrightarrow{\overline{K}N} P}$$

$$\textsc{Case}\ \frac{\Psi \rhd P_i \xrightarrow{\alpha} P' \qquad \Psi \vdash \varphi_i}{\Psi \rhd \mathbf{case}\ \widetilde{\varphi} : \widetilde{P} \xrightarrow{\alpha} P'} \qquad \textsc{Par}\ \frac{\Psi_Q \otimes \Psi \rhd P \xrightarrow{\alpha} P'}{\Psi \rhd P \mid Q \xrightarrow{\alpha} P' \mid Q}\ \mathrm{bn}(\alpha)\#Q$$

$$\textsc{Com}\ \frac{\begin{array}{c}\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \leftrightarrow K \\ \Psi_Q \otimes \Psi \rhd P \xrightarrow{\overline{M}(\nu\widetilde{a})N} P' \qquad \Psi_P \otimes \Psi \rhd Q \xrightarrow{\underline{K}\,N} Q'\end{array}}{\Psi \rhd P \mid Q \xrightarrow{\tau:p} (\nu\widetilde{a})(P' \mid Q')}\ \widetilde{a}\#Q$$

$$\textsc{Rep}\ \frac{\Psi \rhd P \mid !P \xrightarrow{\alpha} P'}{\Psi \rhd !P \xrightarrow{\alpha} P'} \qquad \textsc{Scope}\ \frac{\Psi \rhd P \xrightarrow{\alpha} P'}{\Psi \rhd (\nu b)P \xrightarrow{\alpha} (\nu b)P'}\ b\#\alpha,\Psi$$

$$\textsc{Open}\ \frac{\Psi \rhd P \xrightarrow{\overline{M}(\nu\widetilde{a})N} P'}{\Psi \rhd (\nu b)P \xrightarrow{\overline{M}(\nu\widetilde{a}\cup\{b\})N} P'}\ \begin{array}{l} b\#\widetilde{a},\Psi,M \\ b \in \mathrm{n}(N)\end{array}$$

Table 1. Structured operational semantics. Symmetric versions of Com and Par are elided. In the rule Com we assume that $\mathcal{F}(P) = (\nu\widetilde{b_P})\Psi_P$ and $\mathcal{F}(Q) = (\nu\widetilde{b_Q})\Psi_Q$ where $\widetilde{b_P}$ is fresh for all of $\Psi, \widetilde{b_Q}, Q, M$ and P, and that $\widetilde{b_Q}$ is similarly fresh. In the rule Par we assume that $\mathcal{F}(Q) = (\nu\widetilde{b_Q})\Psi_Q$ where $\widetilde{b_Q}$ is fresh for Ψ, P and α. In Open the expression $\tilde{a} \cup \{b\}$ means the sequence $\tilde{a}$ with b inserted anywhere.

Definition 3 (Transitions). *A* transition *is written $\Psi \rhd P \xrightarrow{\alpha} P'$, meaning that in the environment Ψ the well-formed agent P can do an α to become P'. The transitions are defined inductively in Table 1. We write $P \xrightarrow{\alpha} P'$ without an assertion to mean $\mathbf{1} \rhd P \xrightarrow{\alpha} P'$, and $\Psi \rhd P \xrightarrow{\alpha}$ to mean $\exists P'.\Psi \rhd P \xrightarrow{\alpha} P'$, and $(\nu\widetilde{c})\Psi \rhd P \xrightarrow{\alpha} P'$ to mean $\widetilde{c}\#\,\mathrm{subject}(\alpha), P$ and $\Psi \rhd P \xrightarrow{\alpha} P'$.*

We will sometimes write $F \rhd P \xrightarrow{\alpha} P'$ to mean that there exists $\widetilde{b_F}$ and Ψ_F such that $F = (\nu\widetilde{b_F})\Psi_F$ and $\Psi_F \rhd P \xrightarrow{\alpha} P'$ where $\widetilde{b_F}$ are fresh in the subject of α and in P.

Agents, frames and transitions are identified by alpha equivalence. In a transition the names in $\mathrm{bn}(\alpha)$ bind into both the action object and the derivative, therefore $\mathrm{bn}(\alpha)$ is in the support of α but not in the support of the transition. This means that the bound names can be chosen fresh, substituting each occurrence in both the object and the derivative.

Definition 4 (Strong bisimulation). *A* strong bisimulation $\mathcal{R}$ *is a ternary relation on assertions and pairs of agents such that $\mathcal{R}(\Psi, P, Q)$ implies*

1. *Static equivalence: $\Psi \otimes \mathcal{F}(P) \simeq \Psi \otimes \mathcal{F}(Q)$; and*

2. *Symmetry:* $\mathcal{R}(\Psi, Q, P)$*; and*
3. *Extension of arbitrary assertion:* $\forall \Psi'.\ \mathcal{R}(\Psi \otimes \Psi', P, Q)$*; and*
4. *Simulation: for all* α, P' *such that* $\Psi \rhd P \xrightarrow{\alpha} P'$ *and* $\mathrm{bn}(\alpha)\#\Psi, Q$*, there exists* Q' *such that* $\Psi \rhd Q \xrightarrow{\alpha} Q'$ *and* $\mathcal{R}(\Psi, P', Q')$*.*

We define $P \mathrel{\dot{\sim}_{\Psi}} Q$ *to mean that there exists a bisimulation* $\mathcal{R}$ *such that* $\mathcal{R}(\Psi, P, Q)$*, and write* $\dot{\sim}$ *for* $\dot{\sim}_{\mathbf{1}}$*.*

Definition 5 (Strong congruence). *We define* $P \sim_{\Psi} Q$ *to mean that for all substitution sequences* σ*,* $P\sigma \mathrel{\dot{\sim}_{\Psi}} Q\sigma$ *holds. We write* $P \sim Q$ *to mean* $P \sim_{\Psi} Q$*.*

3 Reliable Broadcast Psi-calculi

Broadcast psi-calculi [6] is an extension of psi-calculi for synchronous unreliable broadcast. A broadcast output action is written $\overline{!K}\ (\nu\widetilde{a})N$, meaning that N with bound names $\widetilde{a}$ is sent along a broadcast channel represented by K. The corresponding (early) broadcast input action is written $\underline{?K}\,N$. The predicates $\dot{\prec}$ and $\dot{\succ}$ regulate how prefix subjects are related to broadcast channels: If $M \mathrel{\dot{\prec}} K$ then an output prefix with subject M will give rise to an action on the broadcast channel K, and similarly $\dot{\succ}$ is used with broadcast input. For technical reasons related to scope extension, no broadcast channel may have greater support than the prefix subjects connected to it. A special case is to declare $K \mathrel{\dot{\prec}} K$ and $K \mathrel{\dot{\succ}} K$ for all broadcast channels K; these channels are then represented by themselves in the process syntax.

The process syntax is the same as for ordinary psi-calculi, but the transition relation is extended with rules to accommodate broadcast communication, as shown in Table 2. The main difference is that broadcast communication does not result in a silent action. To accommodate multiple receivers, it instead yields a broadcast output action which may be picked up by others: $\Psi \rhd \overline{K}\,N\,.\,P \mid \underline{K}(\lambda\widetilde{x})X\,.\,Q \xrightarrow{\overline{!K}\,N} P \mid Q[\widetilde{x} := \widetilde{L}]$, assuming $N = X[\widetilde{x} := \widetilde{L}]$ and $\Psi \vdash K \mathrel{\dot{\prec}} K$ and $\Psi \vdash K \mathrel{\dot{\succ}} K$. An extensive treatment of broadcast psi-calculi including motivations and examples is in [6].

Here broadcast messages can be non-deterministically lost, a behaviour which is appropriate for the intended application area of wireless networks. However, if we wish to study broadcast communication on a shared bus, message loss is not a concern: any component that listens when a message is sent will receive it.

In this section we now define reliable broadcast psi-calculi, a conservative extension of broadcast psi-calculi where negative premises in the Par rule for parallel composition are used to ensure that messages cannot be lost.

One new psi-calculus parameter is added: the equivariant operator $\mathtt{reliable} : \mathbf{T} \to \mathbf{C}$. Intuitively, if $\Psi \vdash \mathtt{reliable}(K)$, then broadcasts over the channel K in the environment Ψ will reach all of its potential recipients. We will refer to such broadcasts as *reliable* broadcasts. Hence, it is possible to have both reliable and unreliable broadcast channels in the calculus, and even have channels whose reliability varies dynamically depending on the environment.

$$\textsc{BrOut}\ \frac{\Psi \vdash M \mathrel{\dot{\prec}} K}{\Psi \rhd \overline{M}\,N\,.\,P \xrightarrow{\overline{!K}\,N} P} \qquad \textsc{BrIn}\ \frac{\Psi \vdash K \mathrel{\dot{\succ}} M}{\Psi \rhd \underline{M}(\lambda\widetilde{y})N\,.\,P \xrightarrow{\underline{?K}\,N[\widetilde{y}:=\widetilde{L}]} P[\widetilde{y}:=\widetilde{L}]}$$

$$\textsc{BrMerge}\ \frac{\Psi_Q \otimes \Psi \rhd P \xrightarrow{\underline{?K}\,N} P' \qquad \Psi_P \otimes \Psi \rhd Q \xrightarrow{\underline{?K}\,N} Q'}{\Psi \rhd P \mid Q \xrightarrow{\underline{?K}\,N} P' \mid Q'}$$

$$\textsc{BrCom}\ \frac{\Psi_Q \otimes \Psi \rhd P \xrightarrow{\overline{!K}\,(\nu\widetilde{a})N} P' \qquad \Psi_P \otimes \Psi \rhd Q \xrightarrow{\underline{?K}\,N} Q'}{\Psi \rhd P \mid Q \xrightarrow{\overline{!K}\,(\nu\widetilde{a})N} P' \mid Q'}\ \widetilde{a}\#Q$$

$$\textsc{BrOpen}\ \frac{\Psi \rhd P \xrightarrow{\overline{!K}\,(\nu\widetilde{a})N} P'}{\Psi \rhd (\nu b)P \xrightarrow{\overline{!K}\,(\nu\widetilde{a}\cup\{b\})N} P'}\ \begin{array}{l} b\#\widetilde{a},\Psi,K \\ b \in \mathrm{n}(N)\end{array}$$

$$\textsc{BrClose}\ \frac{\Psi \rhd P \xrightarrow{\overline{!K}\,(\nu\widetilde{a})N} P'}{\Psi \rhd (\nu b)P \xrightarrow{\tau:p} (\nu b)(\nu\widetilde{a})P'}\ \begin{array}{l} b \in \mathsf{n}(K) \\ b\#\Psi\end{array}$$

Table 2. Additional rules for broadcast psi-calculi. A symmetric version of BrCom is elided. In rules BrCom and BrMerge we assume that $\mathcal{F}(P) = (\nu\widetilde{b_P})\Psi_P$ and $\mathcal{F}(Q) = (\nu\widetilde{b_Q})\Psi_Q$ where $\widetilde{b_P}$ is fresh for $P, \widetilde{b_Q}, Q, K$ and Ψ, and that $\widetilde{b_Q}$ is fresh for $Q, \widetilde{b_P}, P, K$ and Ψ. In BrOpen the expression $\tilde{a} \cup \{b\}$ means the sequence $\tilde{a}$ with b inserted anywhere.

The effects on the semantics are on the Par rule that says, roughly, that any process can always act alone. Eliding the assertions that define the environment, Par means that if $P \xrightarrow{\alpha} P'$ then $P \mid Q \xrightarrow{\alpha} P' \mid Q$. If α is a reliable broadcast that can be received by Q this rule no longer applies. In that case Q must participate in a communication through another rule, involving both P and Q. In order to prevent the Par rule from being applicable in that case we introduce a predicate ADMIT in its premise. Intuitively, ADMIT(α, F, Q) is true if a process in parallel to Q may perform the action α in frame F without Q having to participate in the action. In other words, it is true unless α is a reliable broadcast that Q can receive in frame F.

$$\mathrm{ADMIT}(\alpha, F, Q) = \begin{cases} \mathit{True} \text{ if } \alpha \text{ is not a broadcast action} \\ \neg(F \otimes \mathcal{F}(Q) \vdash \texttt{reliable}(M) \wedge F \rhd Q \xrightarrow{\underline{?M}\,N}) \\ \qquad \text{if } \alpha = \underline{?M}\,N \text{ or } \alpha = \overline{!M}\,(\nu\widetilde{a})N. \end{cases}$$

The only change to the semantics is that the ADMIT predicate is added as a side-condition to the Par rule, as follows. A symmetric version is elided, and to write out the rule in full with the assertions we assume that $\mathcal{F}(Q) = (\nu\widetilde{b_Q})\Psi_Q$ where $\widetilde{b_Q}$ is fresh for Ψ, P and α.

$$\textsc{Par}\ \frac{\Psi_Q \otimes \Psi \,\rhd\, P \ \xrightarrow{\alpha}\ P'}{\Psi \,\rhd\, P \mid Q \ \xrightarrow{\alpha}\ P' \mid Q}\ \begin{array}{l}\mathrm{ADMIT}(\alpha, \Psi \otimes \mathcal{F}(P), Q)\\ \mathrm{bn}(\alpha) \# Q\end{array}$$

Note that the premise of the Par rule is a negative premise since ADMIT requires the absence of transitions.

Finally, a syntactic restriction on processes is imposed: in the agent $!P$, we require that P must be *reliable reception guarded*, which intuitively means that there is no substitution and environment that can make an unguarded prefix into a reliable broadcast input. Formally, an agent is reliable reception guarded if each input is either guarded or its subject M satisfies the following, where σ ranges over substitution sequences:

$$\forall \Psi, N, \sigma. \neg(\Psi \vdash N \,\dot{\succ}\, M\sigma \wedge \Psi \vdash \texttt{reliable}(N))$$

This criterion is imposed in order to ensure that the ruleset is consistent. As an example to demonstrate why it is necessary, consider an agent P with an unguarded reliable broadcast input. The only SOS rule for $!P$ says that it should behave as $P \mid !P$. Now assume that $!P$ has no broadcast input, then by the new Par rule $P \mid !P$ and hence also $!P$ has a broadcast input, leading to a contradiction. On the other hand, if $!P$ has a broadcast input then a shorter inference of that action must have been made from $P \mid !P$; this means that it must have been inferred from P and Par, an impossibility if $!P$ has that action. In other words, it seems that $!P$ has the broadcast action if and only if it has not!

The closure under substitutions in the criterion is needed since $!P$ may occur under an input prefix which, when executing, will give rise to a substitution. Formally, the rule system on agents which are not reliable reception guarded is not stratifiable and thus does not give rise to a meaningful semantics. We do not believe that reliable broadcast input under replication is meaningful — we cannot conceive of a semantics which allows it, while also being faithful to the standard algebraic properties of replication and our intuitive understanding of reliability.

Example 6 (Monadic $b\pi$ calculus). $b\pi$ [10] is an adaptation of the pi-calculus to use broadcast communication in place of point-to-point communication. Here we show a psi-calculus **BPI** that corresponds to monadic $b\pi$. For convenience, we consider only the finite subcalculus, without the τ prefix (since it can be encoded as $(\nu x)\overline{x}\,x$).

BPI	
$\mathbf{T} = \mathcal{N}$	$a \,\dot{\leftrightarrow}\, b = \bot$
$\mathbf{C} = \{\top, \bot\} \cup \{a \dot{=} b \mid a, b \in \mathcal{N}\} \cup \{\neg\phi \mid \phi \in \mathbf{C}\}$	$a \,\dot{\succ}\, b = a \,\dot{\prec}\, b = a \dot{=} b$
$\mathbf{A} = \{\mathbf{1}\}$	$\mathbf{1} \vdash \top$
$\texttt{reliable}(M) = \top$	$\mathbf{1} \vdash a \dot{=} a$
	$\mathbf{1} \vdash \neg\phi$ iff $\neg\mathbf{1} \vdash \phi$

We define representation functions $[\![\cdot]\!]$ from $b\pi$ agents and actions to **BPI** agents and actions, respectively:

Definition 7 ($b\pi$ to PPi).
Agents:

$$\begin{aligned}
[\![P + Q]\!] &= \textbf{case}\ \top : [\![P]\!]\ [\!]\ \top : [\![Q]\!] \\
[\![\langle x = y\rangle P, Q]\!] &= \textbf{case}\ x \dot{=} y : [\![P]\!]\ [\!]\ \neg x \dot{=} y : [\![Q]\!] \\
[\![x(y).P]\!] &= \underline{x}(\lambda y)y.[\![P]\!] \\
[\![\overline{x}y.P]\!] &= \overline{x}\langle y\rangle.[\![P]\!] \\
[\![\text{nil}]\!] &= \mathbf{0} \\
[\![P \mid Q]\!] &= [\![P]\!] \mid [\![Q]\!] \\
[\![\nu x P]\!] &= (\nu x)[\![P]\!]
\end{aligned}$$

Actions:

$$\begin{aligned}
[\![\nu x \overline{a} x]\!] &= \overline{!a}\ (\nu x)x \\
[\![\overline{a}x]\!] &= \overline{!a}\ x \\
[\![x\langle y\rangle]\!] &= \underline{?x}\ y \\
[\![\tau]\!] &= \tau
\end{aligned}$$

Lemma 8. *If $P \xrightarrow{a:}$ then* $\text{ADMIT}(\underline{?a}\,x, \mathbf{1}, [\![P]\!])$.

Proof. By induction on the derivation of $P \xrightarrow{a:}$.

Lemma 9. *If* $\text{ADMIT}(\underline{?a}\,x, \mathbf{1}, [\![P]\!])$ *then $P \xrightarrow{a:}$.*

Proof. By induction on P.

Theorem 10 (Operational correspondence).

1. *If $P \xrightarrow{\alpha} P'$ then $\mathbf{1} \rhd [\![P]\!] \xrightarrow{[\![\alpha]\!]} [\![P']\!]$.*
2. *If $\mathbf{1} \rhd [\![P]\!] \xrightarrow{[\![\alpha]\!]} P'$ and $\text{bn}(\alpha)\#P$, then there exists P'' such that $P \xrightarrow{\alpha} P''$ and $P' = [\![P'']\!]$.*

Proof. By induction on the derivation of the transitions of P and $[\![P]\!]$, respectively.

There can be no similar correspondence in the polyadic case, since structural equivalence (swapping of outermost restrictions, in particular) is not sound for strong labelled bisimilarity in $b\pi$ (in contradiction to [10, Lemma 34(i)]): $\nu x\, \nu y\, \overline{a}\, x, y.\text{nil} \xrightarrow{\nu x\, \nu y\, \overline{a}\, x,y}$ nil, but the only transition of $\nu y\, \nu x\, \overline{a}\, x, y.\text{nil}$ is $\nu y\, \nu x\, \overline{a}\, x, y.\text{nil} \xrightarrow{\nu y\, \nu x\, \overline{a}\, x,y}$ nil, which has a different label.

$$\textsc{Case}\ \frac{\Psi \rhd P_i \xrightarrow{\alpha} P' \qquad \Psi \vdash \varphi_i}{\Psi \rhd \mathbf{case}\ \widetilde{\varphi} : \widetilde{P} \xrightarrow{\alpha} P'}\ \mathrm{HIGHEST}(\alpha, \Psi, \mathbf{case}\ \widetilde{\varphi} : \widetilde{P})$$

$$\textsc{Par}\ \frac{\Psi_Q \otimes \Psi \rhd P \xrightarrow{\alpha} P'}{\Psi \rhd P \mid Q \xrightarrow{\alpha} P' \mid Q}\ \begin{array}{l}\mathrm{HIGHEST}(\alpha, \Psi, P \mid Q)\\ \mathrm{bn}(\alpha) \# Q\end{array}$$

$$\textsc{Com}\ \frac{\begin{array}{c}\Psi \otimes \Psi_P \otimes \Psi_Q \vdash M \dot{\leftrightarrow} K \qquad \Psi \vdash \mathtt{prio}(M) = p\\ \Psi_Q \otimes \Psi \rhd P \xrightarrow{\overline{M}(\nu\widetilde{a})N} P'\\ \Psi_P \otimes \Psi \rhd Q \xrightarrow{\underline{K}\,N} Q'\end{array}}{\Psi \rhd P \mid Q \xrightarrow{\tau : p} (\nu\widetilde{a})(P' \mid Q')}\ \begin{array}{l}\mathrm{HIGHEST}(\tau : p, \Psi, P \mid Q)\\ \widetilde{a} \# Q\end{array}$$

Table 3. Structured operational semantics with priorities. Symmetric versions of Com and Par are elided. In the rule Com we assume that $\mathcal{F}(P) = (\nu\widetilde{b_P})\Psi_P$ and $\mathcal{F}(Q) = (\nu\widetilde{b_Q})\Psi_Q$ where $\widetilde{b_P}$ is fresh for all of $\Psi, \widetilde{b_Q}, Q, M$ and P, and that $\widetilde{b_Q}$ is similarly fresh. In the rule Par we assume that $\mathcal{F}(Q) = (\nu\widetilde{b_Q})\Psi_Q$ where $\widetilde{b_Q}$ is fresh for Ψ, P and α.

4 Psi-calculi with priorities

In this section, we extend psi-calculi with a priority system. The idea is that for each communication channel M, the assertion environment associates to it a priority level $n \in \mathbb{N}$, where lower values of n denote *higher* priority levels. A process may only interact on M if for all $m < n$, no internal actions at priority m are available. As the assertion environment changes, priority levels may change depending on the particulars of the psi-calculus under consideration.

To achieve this, we add the equivariant operator $\mathtt{has\_prio} \in \mathbf{T} \times \mathbb{N} \rightarrow \mathbf{C}$, intuitively $F \vdash \mathtt{has\_prio}(M, p)$ means that the frame F assigns priority p to the term M. If M is a communication channel, i.e., for some K it holds that $\Psi \vdash M \dot{\leftrightarrow} K$, then we require this p to be unique and to be invariant under channel equivalence. We write $\mathtt{prio}(M) = p$ for $\mathtt{has\_prio}(M, p)$. We tag the silent action with an explicit priority, as in $\tau : p$. We notate the priority of an action α in frame F as $\mathtt{PRIO}\,(F, \alpha)$, defined to be p if $\alpha = \tau : p$ or if M is the subject of α and $F \vdash \mathtt{prio}(M) = p$.

To define a transition system with priorities, we use a predicate $\mathrm{HIGHEST}(\alpha, \Psi, P)$, that intuitively states that P has no transitions that block α in the current frame. Transitions that block α in frame F are internal actions with higher priority:

$$\mathrm{HIGHEST}(\alpha, \Psi, P) := \neg(\Psi \rhd P \xrightarrow{\tau : n} \wedge\, n < \mathtt{PRIO}\,(\Psi \otimes \mathcal{F}(P), \alpha))$$

To enforce that transitions respect the priority scheme, HIGHEST is added as a side condition to relevant rules, as in Table 3.

Since HIGHEST requires absence of transitions, it constitutes a negative premise.

Example 11 (Pi-calculus with dynamic priorities). We here define a calculus based on the pi-calculus, with the addition that channels may have one of two priority levels: 0 (high) and 1 (low). Further, it is possible to flip the priority of a channel a dynamically by asserting $\{a\}$. For an example, suppose we want to enforce a fairness scheme such that synchronisations on two channels x and y are guaranteed to interleave. This can be achieved by swapping the priorities of x and y after every such synchronisation, as in the following derivation sequence, where $P_z = (\!|\{z\}|\!) \mid !\overline{x}.(\!|\{x, y\}|\!) \mid !\overline{y}.(\!|\{x, y\}|\!)$.

$$\begin{aligned} \mathbf{1} \rhd\ P_y \mid x\,.\,x\,.\,x \mid y &\xrightarrow{\tau:0} P_x \mid x\,.\,x \mid y \xrightarrow{\tau:0} \\ P_y \mid x\,.\,x &\xrightarrow{\tau:0} P_x \mid x \xrightarrow{\tau:1} P_y \end{aligned}$$

Note that the above τ sequence is the only possible τ sequence — as long as both x and y are available they are guaranteed to be consumed alternatingly. Formally, we define this psi-calculus by $\mathbf{T} = \mathcal{N}$, $\mathbf{C} = \{x = y : x, y \in \mathbf{T}\} \cup \{\texttt{prio}(M) = n : M \in \mathbf{T} \wedge n \in \mathbb{N}\}$ and $\mathbf{A}$ is the finite sets of names. Moreover, let $\mathbf{1}$ be the empty set and $A \otimes B = (A \cup B) - (A \cap B)$. Entailment is defined so that $\Psi \vdash x = y$ iff $x = y$, $\Psi \vdash \texttt{prio}(x) = 1$ iff $x \in \Psi$, and $\Psi \vdash \texttt{prio}(x) = 0$ iff $x \notin \Psi$. Finally, we let channel equivalence be syntactic equality on names.

5 Examples of Reliable Broadcast Priorities

In this section, we combine the extensions for priority and broadcast presented in Section 3 and Section 4 into a single calculus. In order to integrate the two, the following adaptions have to be made. First, high-priority broadcast actions are considered to block lower priority actions, in the same way as high-priority tau actions. The intuition is that both broadcast output and tau are independent actions that can occur without a communication partner. Formally, we let β range over broadcast outputs and τ actions, and amend the definition of HIGHEST as follows:

$$\begin{aligned} \mathrm{HIGHEST}(\alpha, \Psi, P) := \neg\exists\beta.\ & \\ & \Psi \rhd P \xrightarrow{\beta} \\ & \wedge \\ & \texttt{PRIO}\,(\Psi \otimes \mathcal{F}(P), \beta) < \texttt{PRIO}\,(\Psi \otimes \mathcal{F}(P), \alpha) \end{aligned}$$

This predicate is added as a side condition to the rules for broadcast communication, in the same manner as the rules for unicast communication.

Second, we require that broadcast channels have a well defined priority, in the same way as unicast channels. In other words we require that if $\Psi \vdash M \dot{\prec} K$ or $\Psi \vdash K \dot{\succ} M$, then there is a unique p such that $\Psi \vdash \texttt{prio}(K) = p$. If $\Psi \vdash M \dot{\prec} K$ we further require that there is a unique p such that $\Psi \vdash \texttt{prio}(M) = p$, that if $\Psi \vdash \texttt{prio}(K) = p'$ holds then $p \leq p'$, and that if $\Psi \vdash M \dot{\prec} L$ then $\Psi \vdash \texttt{prio}(L) = p'$. These requirements ensure that no actions that may arise from an output prefix will block another action from the same prefix. Note that input prefixes, unlike output prefixes, may be simultaneously connected to broadcast channels with different priorities. Hence, a listener need not care about the priority of messages it receives.

5.1 Discrete Time

There exist several versions of process algebras with discrete time; a common approach is to introduce a special kind of action σ to represent that time passes [15,18]. We claim that in psi-calculi, σ is merely a special case of a reliable broadcast of a clock pulse whose priority is lower than all other actions.

As an example, in the timed broadcasting process calculus aTCWS, due to Macedonio and Merro [18] the intuition is that when no process currently wishes to send anything, a timeout event is propagated through the system. Input prefixes are of the form $\lfloor ?(x).P \rfloor Q$ - this agent may receive a broadcast carrying the message w and evolve to $P\{w/x\}$, or a σ and evolve to Q.

We can easily formulate a psi-calculus to accommodate this. Let σ be a new term such that in all assertion environments $\sigma \mathrel{\dot{\prec}} \sigma \mathrel{\dot{\succ}} \sigma$, $\mathtt{reliable}(\sigma)$ and $\forall \Psi, n, n', M. M \neq \sigma \;\wedge\; \Psi \vdash \mathtt{has\_prio}(M, n) \wedge \Psi \vdash \mathtt{has\_prio}(\sigma, n') \Rightarrow n' \geq n$ holds. Let the system contain a timeout factory process $!\overline{\sigma}\,.$ The aTCWS input prefix $\lfloor ?(x).P \rfloor Q$ can then be represented as $?(x).P + \underline{\sigma}.Q$.

As another example consider a more general timeout operator. Nicollin and Sifakis define in their survey [20] a *timeout* for P (the body), Q (the exception), d (the integer time delay) to behave as P if an initial action of P is performed within time d, otherwise it behaves as Q after time d. Varieties of this operator exist in many process algebras; in e.g. TPCCS [14] it is notated $P \rhd_d Q$.

In a psi-calculus we can represent timeout as follows. Extend the terms to contain $\mathtt{raise}_a$ and $\mathtt{abort}_a$ with support $\{a\}$ for all names a, and extend the broadcast relations such that for all a, Ψ, $\mathtt{raise}_a \mathrel{\dot{\prec}} \mathtt{raise}_a \mathrel{\dot{\succ}} \mathtt{raise}_a$, $\mathtt{reliable}(\mathtt{raise}_a)$, and similarly for $\mathtt{abort}_a$, and with no other channel equivalences on these terms. The idea in the following representation is that an action on $\overline{\mathtt{raise}_a}$ will trigger the exception Q, and that P can at any prior time abort the timeout by performing an action $\overline{\mathtt{abort}_b}$, where a and b are distinct fresh names. Let $\mathtt{raise}_a$ carry a priority higher than any other used in P or Q.

Let a timer T^d for any integer d be defined by $T^0 = \overline{\mathtt{raise}_a}\,.\,\mathbf{0}$ and $T^d = \underline{\sigma}\,.\,T^{d-1} + \underline{\mathtt{abort}_b}\,.\,\mathbf{0}$ for $d > 0$. The timeout $P \rhd_d Q$ then corresponds to

$$(\nu\, a, b)(T^d \mid ((P + \underline{\mathtt{raise}_a}\,.\,\mathbf{0}) \mid (\underline{\mathtt{abort}_b}\,.\,\mathbf{0} + \underline{\mathtt{raise}_a}\,.\,Q)))$$

To examine its behaviour first assume $d > 0$. If $P \xrightarrow{\underline{\sigma}} P'$, i.e, if P can let time pass to become P', then $P \rhd_d Q \xrightarrow{\underline{\sigma}} P' \rhd_{d-1} Q$. If P instead starts acting by aborting the timeout with $P \xrightarrow{\overline{\mathtt{abort}_b}} P'$ then $P \rhd_d Q \xrightarrow{\tau} (\nu\, a, b)(\mathbf{0} \mid P' \mid \mathbf{0})$ which will behave as P' since a, b are chosen fresh. If $d = 0$ the system becomes

$$(\nu\, a, b)(\overline{\mathtt{raise}_a}\,.\,\mathbf{0} \mid ((P + \underline{\mathtt{raise}_a}\,.\,\mathbf{0}) \mid (\underline{\mathtt{abort}_b}\,.\,\mathbf{0} + \underline{\mathtt{raise}_a}\,.\,Q)))$$

which has a broadcast along $\mathtt{raise}_a$ to become $(\nu\, a, b)(\mathbf{0} \mid \mathbf{0} \mid Q)$, which will behave as Q.

There are several other operators related to time which may merit investigation in the same way, and it would be interesting to explore their algebraic properties. For example, it is reasonable to expect that timeout is associative.

5.2 Controller Area Network Bus

The CAN bus [9] is a communication bus designed for system communication in vehicles. It has also been used in other areas than automobiles, such as automation and aerospace. CAN is a message based protocol where all messages are broadcast to everyone. Each node can transmit at any time on the bus. When several nodes want to transmit at the same time, an arbitration-free algorithm based on priorities is used. The identity of each node is an integer, which is used as the priority of the node. A lower identity has a higher priority.

Using a reliable broadcast psi-calculus, we model the protocol used for deciding who has the highest priority to transmit. In CAN, the binary representation of identities are used to decide who gets priority. A 0 is termed a "dominant" bit, and a 1 is termed a "recessive" bit. A dominant bit will overwrite a recessive bit on the bus. When several nodes want to transmit at the same time, they transmit their identities bit by bit, while listening on the bus to see if their bit is dominated by other nodes. If any node has a dominant bit, all other nodes will see it. The nodes that have a recessive bit will see the dominant bit on the bus, back off, and attempt transmission at a later time.

In order to formulate a model we begin by defining a suitable psi-calculus, by instantiating the parameters to our framework. For our terms, we use names, a set of constants for channels and binary numbers, and a list construction, so that we can represent binary sequences. **bus**, **can**, σ_1 and σ_2 are all reliable broadcast channels. Priorities are set as follows: $(\sigma_1, 3), (\sigma_2, 1), (\mathbf{bus}, 0), (\mathbf{can}, 0)$, and for all names a, $(a, 2)$. The resulting instance is as follows.

Psi-calculus for CAN

$\mathbf{T} = \mathcal{N} \cup \{\mathbf{bus}, \mathbf{can}, \sigma_1, \sigma_2, 0, 1\} \cup \{M\#N : M, N \in T\}$
$\mathbf{C} = \{\top, \bot\}$
$\mathbf{A} = \{\mathbf{1}\}$
$M \leftrightarrow N = \top$ if $M = N \in \mathcal{N}$, otherwise $\bot$
$M \dot{\succ} N = M \dot{\prec} N = \top$ if $M = N \in \{\mathbf{bus}, \mathbf{can}, \sigma_1, \sigma_2\}$, otherwise $\bot$
$\mathtt{reliable}(M) = \top$
$\text{hasprio}(M, n) = \top$ iff $(M, n) \in \{(\sigma_1, 3), (\sigma_2, 1), (\mathbf{bus}, 0), (\mathbf{can}, 0)\}$
$\qquad \cup \{(a, 2) : a \in \mathcal{N}\}$
$\mathbf{1} \vdash \top$

A node executing the arbitration protocol is written $F(tl, id, M)$. This represents a node with identity id wanting to transmit M on the bus. The process is defined recursively, and tl is what remains of id to be transmitted. σ_1 and σ_2 are clock channels. σ_1 regulates the cycle of arbitration sessions, and σ_2 regulates the cycle of bit arbitration within a session. **bus** is the channel used for arbitrations, and **can** is the channel for data transmission.

$$\begin{aligned} F(\epsilon, id, M) &= \overline{\mathbf{can}}\, M \\ F(0 :: tl, id, M) &= \overline{\mathbf{bus}}.F(tl, id, M) + \underline{\mathbf{bus}}.F(tl, id, M) \\ F(1 :: tl, id, M) &= \underline{\mathbf{bus}}.\underline{\sigma_1}.F(id, id, M) + \underline{\sigma_2}.F(tl, id, M) \end{aligned}$$

The base case for the recursion is $F(\epsilon, id, M)$. This simulates the case where all arbitration bits have been consumed, and the node has not been dominated. This means that it is the last node standing in the arbitration process, and therefore gets to send its message on the **can** bus. $F(0 :: tl, id, M)$ simulates the case of the node having a dominant arbitration bit. In this case, one of the nodes with a dominant bit will send it on **bus**, and the rest will receive it. A node with a dominant bit will always progress to $F(tl, id, M)$, and thus stay in the arbitration process. $F(1 :: tl, id, M)$ simulates the case of the node having a recessive arbitration bit. Since **bus** has higher priority than σ_2, then if another node is dominant, the recessive node will see the dominant node's transmission on **bus**, back off and wait for a signal on σ_1 to indicate the beginning of the next arbitration session. If there is no dominant node, and thus no transmission on **bus**, σ_2 can synchronise and allow all recessive nodes to enter the next cycle in the arbitration process.

A system where for all $i \leq n$, the node with identity id_i wishes to send M_i can be given as:

$$F(id_0, id_0, M_0) \mid \ldots \mid F(id_n, id_n, M_n) \mid !\overline{\sigma_2} \mid !\overline{\sigma_1}$$

We show next an example of two nodes $F(1010, 1010, M_{1010})$ and $F(1000, 1000, M_{1000})$ competing for the right to transmit their message on **can**. Priority annotations are included to help readability, but are not part of the action syntax of the framework:

$$\begin{aligned} &\underline{\mathbf{bus}}.\underline{\sigma_1}.F(1010, 1010, M_{1010}) + \underline{\sigma_2}.F(010, 1010, M_{1010}) \\ \mid\ &\underline{\mathbf{bus}}.\underline{\sigma_1}.F(1000, 1000, M_{1000}) + \underline{\sigma_2}.F(000, 1000, M_{1000}) \mid !\overline{\sigma_2} \mid !\overline{\sigma_1} \end{aligned}$$

Since both are recessive, neither receives anything on **bus**. The system instead does a σ_2 to reveal another bit:

$$\begin{aligned} \xrightarrow{!\overline{\sigma_2}:1}\ &\overline{\mathbf{bus}}.F(10, 1010, M_{1010}) + \underline{\mathbf{bus}}.F(10, 1010, M_{1010}) \\ \mid\ &\overline{\mathbf{bus}}.F(00, 1000, M_{1000}) + \underline{\mathbf{bus}}.F(00, 1000, M_{1000}) \mid !\overline{\sigma_2} \mid !\overline{\sigma_1} \end{aligned}$$

Since both are now dominant, either might choose to signal on **bus**, but the other will continue to persist:

$$\begin{aligned} \xrightarrow{!\overline{\mathbf{bus}}:0}\ &\underline{\mathbf{bus}}.\underline{\sigma_1}.F(1010, 1010, M_{1010}) + \underline{\sigma_2}.F(0, 1010, M_{1010}) \\ \mid\ &\overline{\mathbf{bus}}.F(0, 1000, M_{1000}) + \underline{\mathbf{bus}}.F(0, 1000, M_{1000}) \mid !\overline{\sigma_2} \mid !\overline{\sigma_1} \end{aligned}$$

Node 1010 is now recessive and node 1000 dominant. Since the σ_2 channel has lower priority than **bus**, the only thing that can happen is a communication on

bus:

$$\begin{aligned}\xrightarrow{!\overline{\mathbf{bus}{:}0}}\ & \underline{\sigma_1}.F(1010, 1010, M_{1010}) \\ & \mid \overline{\underline{\mathbf{bus}}}.F(\epsilon, 1000, M_{1000}) + \underline{\mathbf{bus}}.F(\epsilon, 1000, M_{1000}) \mid !\overline{\sigma_2} \mid !\overline{\sigma_1}\end{aligned}$$

A reliable broadcast does not need any recipients to synchronise with. Thus, node 1000 dominates the bus on its own and progresses to:

$$\begin{aligned}\xrightarrow{!\overline{\mathbf{bus}{:}0}}\ & \underline{\sigma_1}.F(1010, 1010, M_{1010}) \\ & \mid \overline{\mathbf{can}}\, M_{1000} \mid !\overline{\sigma_2} \mid !\overline{\sigma_1}\end{aligned}$$

Finally, since **can** has higher priority than σ_1, M_{1000} is transmitted. After that, a σ_1 transition can proceed to let node 1010 try again.

In [17], Jan and Zdenek models CAN using the graphical formalism of timed automata and verifying properties in UPPAAL. Their model of the arbitration protocol is similar to ours. In [29], Osch and Smolka models and verifies CAN using Murϕ. At a bit over 100 lines, their model of the arbitration protocol is less concise than ours. We have the benefit of modelling the protocol in a bespoke language, defined as an instance of the psi-calculi framework, where the salient features of the protocol have a direct representation. This language may also be extended and formally coexist with arbitrary data structures for data transmitted along the bus, and inherits a machine checked meta-theory.

6 Meta-theory

In this section we discuss the meta-theory of our calculi, including negative premises, algebraic properties of bisimulation, and our Isabelle formalisation.

6.1 Negative Premises

The predicates ADMIT and HIGHEST are used as negative premises. Here, the SOS rules defining the transition relation $\longrightarrow$ are consistent. We show this by constructing a *stratification* S from transition judgments to a well-ordered set such that for every instance of a rule application that may occur in a derivation tree, the negative premises have smaller S-values than the conclusion, and the positive premises have no larger S-values than the conclusion. This guarantees that there are no transitions whose absence is a precondition for their presence.

Definition 12. *We let* $S(\Psi \rhd P \xrightarrow{\alpha} P')$ *be the pair* $(\mathtt{PRIO}\,(\Psi \otimes \mathcal{F}(P), \alpha), s)$*, where* s *is the syntactic size of* P *if* α *is a broadcast input with* $\Psi \otimes \mathcal{F}(P) \vdash \mathtt{reliable}(\mathrm{subj}(\alpha))$ *and* ω *otherwise, and comparisons use the lexical ordering.*

Theorem 13. *S is a stratification of the transition system $\longrightarrow$.*

Proof. By case analysis, using the fact that HIGHEST only considers transitions on strictly lower strata. In the case of REP, we note that α cannot be a reliable broadcast input action since P is reliable reception guarded—hence both premise and conclusion are on stratum (p, ω) for some p. The other cases are trivial.

The transition system $\longrightarrow$ is defined by constructing transitions stratum by stratum in a bottom-up fashion, as explained in the introduction. Further, Groote shows that with this method, any two valid stratifications of a transition system specification yield the same transition relation: hence the stratification can be elided. In our Nominal Isabelle formalisation discussed in Section 6.2, we do not directly employ negative premises, in order to avoid defining the semantics by induction over strata.

We give an equivalent alternative definition inspired by the notions of discard relations [11] and initial actions [16].

Definition 14. *Define predicate* $\mathrm{HIGHEST}_{\mathrm{U}}$ *as* $\mathrm{HIGHEST}$ *with all instances of the transition relation* $\longrightarrow$ *replaced by* $\longrightarrow_{\mathrm{U}}$*, and similarly for* ADMIT*, where* $\longrightarrow_{\mathrm{U}}$ *is* $\longrightarrow$ *with all negative premises removed from the rules. We then define a new transition system* $\longrightarrow_{\mathrm{I}}$ *by replacing all uses of* $\mathrm{HIGHEST}$ *in the SOS rules for* $\longrightarrow$ *by* $\mathrm{HIGHEST}_{\mathrm{U}}$ *(and similarly for* ADMIT*).*

Note that $\longrightarrow_{\mathrm{I}}$ does not contain any negative premises in the formal sense, so it is more readily formalized in Isabelle. Moreover, it coincides with $\longrightarrow$.

Lemma 15. $\mathrm{HIGHEST}_{\mathrm{U}}(\alpha, \Psi, P) \iff \mathrm{HIGHEST}_{\mathrm{I}}(\alpha, \Psi, P)$

Proof. This lemma has been formally proven in Nominal Isabelle.

Lemma 16. $\mathrm{HIGHEST}_{\mathrm{U}}(\alpha, \Psi, P \mid Q) \implies$
$(\mathrm{ADMIT}_{\mathrm{U}}(\alpha, \Psi, P, Q) \iff \mathrm{ADMIT}_{\mathrm{I}}(\alpha, \Psi, P, Q))$.

Proof. This lemma has been formally proven in Nominal Isabelle.

Theorem 17. $\longrightarrow\, = \,\longrightarrow_{\mathrm{I}}$

Proof (sketch). We consider transitions $\Psi \rhd R \xrightarrow{\alpha}_{*} R'$ where $*$ is nothing or I. The proof is by strong induction on the priority p of the action α in frame $\mathcal{F}(R) \otimes \Psi$. Here $\mathrm{HIGHEST}(\alpha, \Psi, R)$ iff (by induction hypothesis) $\mathrm{HIGHEST}_{\mathrm{I}}(\alpha, \Psi, R)$ iff (by Lemma 15) $\mathrm{HIGHEST}_{\mathrm{U}}(\alpha, \Psi, R)$. We proceed by structural induction on the origin R of the transition. Most cases for R follow by case analysis on the topmost derivation rule; the remaining are as follows.

$R = P \mid Q$ **using** RPAR**:** By induction P has the same transitions with $\longrightarrow$ as with $\longrightarrow_{\mathrm{I}}$. We have $\mathrm{HIGHEST}(\alpha, \Psi, R) \iff \mathrm{HIGHEST}_{\mathrm{U}}(\alpha, \Psi, R)$ by the outer induction. Remains the side condition ADMIT.
- $\Rightarrow$ Assume that $\mathrm{ADMIT}(\alpha, \Psi, P, Q)$. By induction we get $\mathrm{ADMIT}_{\mathrm{I}}(\alpha, \Psi, P, Q)$. By Lemma 16 $\mathrm{ADMIT}_{\mathrm{U}}(\alpha, \Psi, P, Q)$.
- $\Leftarrow$ Assume that $\mathrm{ADMIT}_{\mathrm{U}}(\alpha, \Psi, P, Q)$. By Lemma 16 $\mathrm{ADMIT}_{\mathrm{I}}(\alpha, \Psi, P, Q)$. By induction $\mathrm{ADMIT}(\alpha, \Psi, P, Q)$.

$R = !P$ We proceed by induction on the number of uses of the REP rule to unfold $!P$ in the derivation of the transition.

Base case Here the transition was derived using REP;RPAR. By induction P has the same transitions in the system $\longrightarrow$ as in $\longrightarrow_{\mathrm{I}}$. Moreover, $\mathrm{ADMIT}(\alpha, \Psi, P, !P)$ and $\mathrm{ADMIT}_{\mathrm{U}}(\alpha, \Psi, P, !P)$ are always satisfied, since P is BI-guarded and $!P$ thus has no broadcast input transitions.

Induction case By case analysis on the rule used to derive the transition from $P \mid !P$; most cases follow directly from the induction hypotheses. The exception is RPar-R. Here, by induction the transition from $!P$ exists in $\longrightarrow$ iff it exists in $\longrightarrow_{\mathrm{I}}$. Moreover, $\mathrm{ADMIT}(\alpha, \Psi, !P, P)$ and $\mathrm{ADMIT}_{\mathrm{U}}(\alpha, \Psi, !P, P)$ are always satisfied since P is BI-guarded.

6.2 Bisimulation

For any reliable broadcast psi-calculus, labelled bisimilarity respects important algebraic laws. Recall the definitions of strong bisimilarity and congruence are from Section 2. Strong bisimulation is preserved by all operators except input prefix and satisfies the expected algebraic laws such as scope extension:

Theorem 18 (Congruence properties of strong bisimulation). *For all Ψ:*

$$\begin{array}{rcl}
P \mathrel{\dot{\sim}_\Psi} Q & \Longrightarrow & P \mid R \mathrel{\dot{\sim}_\Psi} Q \mid R \\
P \mathrel{\dot{\sim}_\Psi} Q & \Longrightarrow & (\nu a)P \mathrel{\dot{\sim}_\Psi} (\nu a)Q \qquad \textit{if } a\#\Psi \\
P \mathrel{\dot{\sim}_\Psi} Q & \Longrightarrow & !P \mathrel{\dot{\sim}_\Psi} !Q \qquad \textit{if } P, Q \textit{ assertion guarded} \\
\forall i. P_i \mathrel{\dot{\sim}_\Psi} Q_i & \Longrightarrow & \mathbf{case}\ \widetilde{\varphi} : \widetilde{P} \mathrel{\dot{\sim}_\Psi} \mathbf{case}\ \widetilde{\varphi} : \widetilde{Q} \\
P \mathrel{\dot{\sim}_\Psi} Q & \Longrightarrow & \overline{M}\, N \,.\, P \mathrel{\dot{\sim}_\Psi} \overline{M}\, N \,.\, Q \\
(\forall \widetilde{L}.\ P[\widetilde{x} := \widetilde{L}] \mathrel{\dot{\sim}_\Psi} Q[\widetilde{x} := \widetilde{L}]) & \Longrightarrow & \underline{M}(\lambda \widetilde{x})N \,.\, P \mathrel{\dot{\sim}_\Psi} \underline{M}(\lambda \widetilde{x})N \,.\, Q
\end{array}$$

Theorem 19. *Strong congruence $\sim_\Psi$ is a congruence for all Ψ.*

The standard rules of structural equivalence are sound for bisimilarity congruence.

Theorem 20 (Structural equivalence). *Assume that $a\#Q, \widetilde{x}, M, N, \widetilde{\varphi}$. Then*

$$\begin{array}{rcl}
\mathbf{case}\ \widetilde{\varphi} : \widetilde{(\nu a)P} \sim (\nu a)\mathbf{case}\ \widetilde{\varphi} : \widetilde{P} & \qquad & (\nu a)\mathbf{0} \sim \mathbf{0} \\
\underline{M}(\lambda \widetilde{x})N \,.\, (\nu a)P \sim (\nu a)\underline{M}(\lambda \widetilde{x})(N) \,.\, P & & Q \mid (\nu a)P \sim (\nu a)(Q \mid P) \\
\overline{M}\, N \,.\, (\nu a)P \sim (\nu a)\overline{M}\, N \,.\, P & & (\nu b)(\nu a)P \sim (\nu a)(\nu b)P \\
P \mid (Q \mid R) \sim (P \mid Q) \mid R & & !P \sim P \mid !P \\
P \mid Q \sim Q \mid P & & P \sim P \mid \mathbf{0}
\end{array}$$

We have formalised reliable broadcast psi-calculi and the proofs of all theorems in Section 6.2 in Nominal Isabelle [27] — the proof scripts are available online [1]. Since considerable effort has been invested in a proof repository for broadcast psi-calculi [26], we strive to re-use as much of it as possible. However, using negative premises and induction over strata would constitute a major change to the structure of the definition of the operational semantics. Instead, we formalise the two semantics $\longrightarrow_{\mathrm{U}}$ and $\longrightarrow_{\mathrm{I}}$ (Definition 14). Using this strategy, we were able to re-use our repository with only minimal changes to the proofs. The proof scripts constitute 28469 lines of code for $\longrightarrow_{\mathrm{I}}$ and 33823 lines for $\longrightarrow_{\mathrm{U}}$, most of it straightforward adaptations of the existing proof scripts.

The bulk of our adaptation efforts concerns proving various preservation properties of the $\mathrm{ADMIT}_{\mathrm{U}}$ predicate used in the structural congruence proofs, such as

$$\mathrm{ADMIT}_{\mathrm{U}}(\alpha, \Psi, F, Q \mid R) \rightarrow \mathrm{ADMIT}_{\mathrm{U}}(\alpha, \Psi, F \otimes \mathcal{F}(R), Q)$$

Other than such details, the proofs are remarkably similar to the case of unreliable broadcast. The same technical lemmas, overarching proof structures and candidate relations are used. Hence, we shall not restate them here. This is a compelling argument for using proof mechanisation: even though the initial effort of developing a proof repository for broadcast psi-calculi was significant, adopting it for the reliable case took only roughly two man-weeks of effort. Adding priorities to unicast psi-calculi took a similar amount of time, as did combining priorities and reliable broadcasts into a single calculus. We consider this a small price to pay for absolute certainty of correctness, at least up to the current state of the art of mechanised proofs.

7 Related work

As early as 1985, Pnueli [22] formulated the semantics of a parallel composition operator with broadcast synchronisation using negative premises. This predates the work on negative premises by Groote, and shows no awareness that introducing such rules may lead to consistency issues. Fortunately, all rules pertaining to it are syntax-directed and hence trivial to stratify. The first account of priorities in process algebra is by Cleaveland and Hennessy [7], who define the semantics using two strata, an idea we also use in defining $\longrightarrow_{\mathrm{U}}$ and $\longrightarrow_{\mathrm{I}}$. A more recent exposition of this area is in [8].

Prasad [23] expresses reliable broadcast in CBS by introducing so-called "lose" actions, alongside the usual input and output, to denote that a process may discard a message. Prasad remarks that "a loss encodes a negative premise", and that "it is possible to formulate CBS too in terms of negative premises". We go further than this by providing both formulations and proving that they correspond.

Prasad also extends CBS with priorities [24]. The priority system is a static one, where every prefix is tagged with an explicit priority level. This yields a very clean operational semantics, with no need for the negative premises or two-stage operational semantics used in the case of unicast communication. Our own formulation is by necessity more complicated since it incorporates features absent in CBS such as dynamic priorities, mobility, unicast communication and the possibility of lossy broadcasts.

The $b\pi$ calculus by Ene and Muntean [10,11] is a version of the pi-calculus featuring reliable broadcast instead of point-to-point communication. In lieu of negative premises, they use an auxiliary "discard" relation to determine whether a process is allowed to discard a message. When compared to Prasad's "lose" actions, this strategy has the benefit of not introducing additional actions into the semantics. Moraru et al. [19] make an initial effort towards an extension of $b\pi$ with ambients and the possibility of encrypting messages.

Gunter and Yasmeen's work on Secure Broadcast Ambients [13] does not feature reliable broadcast, but is another example of a broadcast calculus implemented in Nominal Isabelle [30]. The authors describe the implementation of the syntax and semantics of their calculus, as well as barbed congruence and equivalence. However, they do not use their framework to prove any theorems.

Jeffrey [16] shows how to translate a variant of timed CCS into prioritised CCS by attaching an explicit time stamp to each action's priority. Crucially, Jeffrey's translation relies on the CSP synchronisation operator, which can be seen as a form of reliable broadcast communication. Prasad achieves a similar feat in [25], where he argues that prioritised CBS is isomorphic to a subset of timed CBS. This, in conjunction with our own findings, leads us to believe that Jeffrey's observation that "time is an example of priority" ought to be clarified: merely having priorities is not sufficient. A one-to-many, non-lossy synchronisation mechanism is also required.

8 Conclusion

Negative premises to formulate reliable broadcast and priorities has a long history. Our contribution is to introduce these ideas in psi-calculi, a general framework where highly specialised modelling languages are easily defined, and where a proof repository for Nominal Isabelle guarantees that important meta-theoretical properties related to bisimulation is sound. We go beyond existing process algebras in that priorities and reliability can vary dynamically as processes execute. We have also investigated the expressivity of reliable broadcast with priorities by modelling discrete timeouts and the arbitration protocols of a vehicular bus.

In future work, we intend to add these features to sorted psi-calculi [5]. We also intend to study the properties of a cache coherence protocol using Nominal Isabelle. It would be interesting to find a more general characterization of which instances of negative premises can be encoded in a similar two-level style. We also want to investigate whether there is a symbolic operational semantics that captures psi-calculi with priorities and reliable broadcast.

References

1. Johannes Åman Pohjola. Reliable broadcast psi-calculus with priorities formalisation. `http://www.it.uu.se/research/group/mobility/theorem/reliablebroadcast.tgz`, April 2013. Isabelle/HOL-Nominal formalisation of the definitions, theorems and proofs.
2. Jesper Bengtson. *Formalising process calculi.* PhD thesis, Uppsala University, June 2010.
3. Jesper Bengtson, Magnus Johansson, Joachim Parrow, and Björn Victor. Psi-calculi: A framework for mobile processes with nominal data and logic. *Logical Methods in Computer Science*, 7(1), 2011.

4. Bard Bloom, Sorin Istrail, and Albert R. Meyer. Bisimulation can't be traced: preliminary report. In *Department of Computer Science, Cornell University*, pages 229–239, 1988.
5. Johannes Borgström, Ramūnas Gutkovas, Joachim Parrow, Björn Victor, and Johannes Åman Pohjola. A sorted semantic framework for applied process calculi (extended abstract), 2013. Submitted; an early version entitled *Sorted Psi-calculi with Generalised Pattern Matching* was presented at ICE'12.
6. Johannes Borgström, Shuqin Huang, Magnus Johansson, Palle Raabjerg, Björn Victor, Johannes Åman Pohjola, and Joachim Parrow. Broadcast psi-calculi with an application to wireless protocols. In Gilles Barthe, Alberto Pardo, and Gerardo Schneider, editors, *Software Engineering and Formal Methods: SEFM 2011*, volume 7041 of *Lecture Notes in Computer Science*, pages 74–89. Springer-Verlag, November 2011.
7. Rance Cleaveland and Matthew Hennessy. Priorities in process algebras. In *LICS*, pages 193–202. IEEE Computer Society, 1988.
8. Rance Cleaveland, Gerald Lüttgen, and V. Natarajan. Priority and abstraction in process algebra. *Inf. Comput.*, 205(9):1426–1458, 2007.
9. Steve Corrigan. Introduction to the controller area network (CAN). Technical Report SLOA101A, Texas Instruments, July 2008.
10. Cristian Ene. *Un Modèle formel pour les systèmes mobiles a diffusion.* Phd thesis, Université de la Méditerranée – Marseille, 2001.
11. Cristian Ene and Traian Muntean. A broadcast-based calculus for communicating systems. In *Proceedings of the 15th International Parallel & Distributed Processing Symposium*, IPDPS '01, pages 149–, Washington, DC, USA, 2001. IEEE Computer Society.
12. Jan Friso Groote. Transition system specifications with negative premises. *Theor. Comput. Sci.*, 118(2):263–299, September 1993.
13. Elsa Gunter and Ayesha Yasmeen. Secure broadcast ambients. In Pierpaolo Degano, Joshua Guttman, and Fabio Martinelli, editors, *Formal Aspects in Security and Trust*, volume 5491 of *Lecture Notes in Computer Science*, pages 257–271. Springer Berlin / Heidelberg, 2009.
14. Hans Hansson and Bengt Jonsson. A calculus for communicating systems with time and probabitilies. In *RTSS*, pages 278–287. IEEE Computer Society, 1990.
15. Matthew Hennessy and Tim Regan. A process algebra for timed systems. *Inf. Comput.*, 117(2):221–239, 1995.
16. Alan Jeffrey. Translating timed process algebra into prioritized process algebra. In Jan Vytopil, editor, *FTRTFT*, volume 571 of *Lecture Notes in Computer Science*, pages 493–506. Springer, 1992.
17. Jan Krakora and Zdenek Hanzálek. Timed automata approach to CAN verification. In P. Kopacek, C.E. Pereira, and G. Morel, editors, *Information Control Problems in Manufacturing*, pages 147–152. IFAC, 2004.
18. Damiano Macedonio and Massimo Merro. A semantic analysis of wireless network security protocols. In Alwyn Goodloe and Suzette Person, editors, *NASA Formal Methods*, volume 7226 of *Lecture Notes in Computer Science*, pages 403–417. Springer, 2012.
19. Victor Moraru, Traian Muntean, and Emilian Gutuleac. Towards a model for broadcasting secure mobile processes. In *Proceedings of The Fifth International Symposium on Parallel and Distributed Computing*, ISPDC '06, pages 168–172, Washington, DC, USA, 2006. IEEE Computer Society.

20. Xavier Nicollin and Joseph Sifakis. An overview and synthesis on timed process algebras. In Kim Guldstrand Larsen and Arne Skou, editors, *CAV*, volume 575 of *Lecture Notes in Computer Science*, pages 376–398. Springer, 1991.
21. A. M. Pitts. Nominal logic, a first order theory of names and binding. *Information and Computation*, 186:165–193, 2003.
22. Amir Pnueli. Linear and branching structures in the semantics and logics of reactive systems. In Wilfried Brauer, editor, *ICALP*, volume 194 of *Lecture Notes in Computer Science*, pages 15–32. Springer, 1985.
23. K. V. S. Prasad. A calculus of value broadcasts. In *IN PARLE'93*, pages 69–4. Springer Verlag LNCS, 1993.
24. K. V. S. Prasad. Broadcasting with priority. In Donald Sannella, editor, *ESOP*, volume 788 of *Lecture Notes in Computer Science*, pages 469–484. Springer, 1994.
25. K. V. S. Prasad. Broadcasting in time. In Paolo Ciancarini and Chris Hankin, editors, *COORDINATION*, volume 1061 of *Lecture Notes in Computer Science*, pages 321–338. Springer, 1996.
26. Palle Raabjerg and Johannes Åman Pohjola. Broadcast psi-calculus formalisation. `http://www.it.uu.se/research/group/mobility/theorem/broadcastpsi`, July 2011. Isabelle/HOL-Nominal formalisation of the definitions, theorems and proofs.
27. Christian Urban and Christine Tasson. Nominal techniques in Isabelle/HOL. In Robert Nieuwenhuis, editor, *Proceedings of CADE 2005*, volume 3632 of *Lecture Notes in Computer Science*, pages 38–53. Springer-Verlag, 2005.
28. Rob J. van Glabbeek. The meaning of negative premises in transition system specifications ii. *J. Log. Algebr. Program.*, 60-61:229–258, 2004.
29. M. J. P. van Osch and Scott A. Smolka. Finite-state analysis of the CAN bus protocol. In *HASE*, pages 42–. IEEE Computer Society, 2001.
30. Ayesha Yasmeen and Elsa L. Gunter. Implementing Secure Broadcast Ambients in Isabelle using Nominal Logic. In *Emerging Trends Report of the 21st International Conference on Theorem Proving in Higher Order Logics (TPHOLs 2008)*, pages 123–134, 2008.