Towards Proofs as Successful Executions of Processes

Ross Horne and Alwen Tiu

School of Computer Science and Engineering, Nanyang Technological University, Singapore
{rhorne,atiu}@ntu.edu.sg

Abstract

We further the understanding of the relationship between process calculi and non-commutative logic. This work focuses on, a first-order extension of the proof calculus BV featuring a de Morgan dual pair of nominal quantifiers, called BV1. An embedding of π-calculus process as predicates in BV1 is defined, and a procedure is provided for extracting successful executions from proofs of embedded processes. This procedure is used to establish the soundness of linear implication in BV1 with respect to trace inclusion in the π-calculus. We illustrate the expressive power of BV1, by demonstrating that these techniques extend also to the internal π-calculus, where privacy of inputs are guaranteed. We emphasise that linear implication is strictly finer than trace inclusion, providing a tight refinement semantics for processes respecting both causality and the scope of private names.

1 Introduction

This paper contributes to a line of work formally connecting proof systems expressed in the calculus of structures [10] and concurrency theory. The calculus of structures is a generalisation of the sequent calculus that in which proof systems can be designed that cannot be expressed in the sequent calculus [26], notably the non-commutative logic BV [10] and its extensions NEL [11, 25] and MAV [12]. Key observations made in this line of work include that, completed executions of processes in a fragment of CCS can be extracted from proofs in BV [6]. A consequence is that linear implication can be shown to be sound with respect to trace equivalence. Elsewhere in the literature, certain homomorphisms over series-parallel pomsets [8] can be used to provide a finer non-interleaving semantics that is sound and complete with respect to linear implication. Indeed BV was motivated by pomset logic [22], so it should be no surprise that linear implication in BV is at the non-interleaving end of the spectrum of preorders over processes.

This paper continues efforts to extend the above observations to more expressive process languages including the π-calculus [20]. A companion paper [13] introduced a first-order extension of MAV, called MAV1, featuring additive operators modelling choice, first-order universal and existential quantifiers and a novel de Morgan dual pair of nominal quantifiers called "new" and "wen". Nominal quantifiers are used to encode fresh names in π-calculus, to avoid two distinct names being identified during refinement. Specifically in the π-calculus $\nu x.\nu z.P$ is in general unrelated to the process $\nu x.P\{^{z}/_{x}\}$. In the later process there is a loss of information compared to the former, since two private quantities, such as nonces of two participants in a protocol are revealed to be the same name. This observation motivated the introduction of the self-dual ∇-quantifier [19]. In our encoding of processes, we further require that nominal quantifiers do not distribute over parallel composition, in contrast to the ∇-quantifier. For example, in the π-calculus a process of the form $\nu x.P \parallel \nu x.Q$ should not in general be related by refinement to $\nu x.(P \parallel Q)$. A cut elimination result for MAV1 has been developed in the companion paper establishing, amongst other consequences, the consistency of MAV1 as a logical system.

For the sake of clarity, here we restrict ourselves to the system BV1, excluding the additive operators that model choice in MAV1. This work clarifies that, for embeddings of processes as predicates, linear implication in BV1 is sound with respect to completed trace equivalence in both the π-calculus and the πI-calculus [23]. As for the fragment of CCS [6], the main technical device is a procedure for extracting successful executions of processes from proofs. The result presented is a minimal result in

terms of situating linear implication in BV1 in the spectrum of preorders over processes. As for BV a tighter semantics should be fundamentally non-interleaving.

Related work. As already mentioned, our work builds on the prior work of Bruscoli [6] in relating BV and CCS. There has been a diverse range of work relating process algebra and logic; we shall discuss briefly those that take the view point of embedding processes as predicates, rather than processes as proofs [1]. In the process-as-predicates embedding approach, Miller's work [18] on embedding π-calculus as a theory in linear logic is perhaps one of the earliest (if not the earliest). In Miller's encoding, input and output prefixes are encoded via non logical constants, whose behaviours are defined via a theory in a higher-order version of linear logic. There the semantics of a process P is defined in terms of the set of its *co-agents* A such that the embedding of $P \parallel A$ in logic is provable. A correspondence is then shown between this semantics definition with that of Hennessy-Milner logic. However, this correspondence was shown only for a fragment of π-calculus without the restriction operator. Other notable related work includes those that consider deep-embedding of transition relations as predicates, such as [17, 24, 27, 4], where the transition relation is either defined as a fixed point definition or logical theories, or even as a special primitive in the logic [24]. We note that in this approach, implication in logic does not immediately gives rise to a pre-order on processes.

Summary. Section 2, recalls and strengthen established results relating BV to a basic process calculus. Section 3 introduces the calculus BV1 and an embedding of π-calculus processes as predicates in BV1. Section 4 is the main technical contribution of the paper. It demonstrates how successful executions of a process can be extracted from the proof of its embedding. From this we are able to establish the soundness of trace inclusion. Section 5 highlights that techniques extended to other process calculi.

2 Background: From Lazy Proofs in BV to Successful Executions

We begin by recalling and elaborating on established results relating the proof calculus BV to a fragment of the process calculus CCS. The established result is that linear implication is sound with respect to completed trace inclusion. This result follows from cut elimination and a decomposition result for structures in BV without an operator called *times*.

We elaborate on this established result by showing a counterexample demonstrating that it is impossible to completely lift the restriction on the syntax of BV forbidding *times*. Despite this this counterexample, we demonstrate that we can define certain propositions where the use of *times* is controlled such that the relevant decomposition is possible. We observe a stronger result: that linear implication is sound and complete with respect to completed pomset trace inclusion.

2.1 The syntax and semantics of BV

Firstly, recall the syntax and semantics of BV. The calculus extends multiplicative linear logic, featuring operators *times* and *par*, with a non-commutative operator called *seq*. There is a single unit I shared by all multiplicative operators.

A derivation in BV is a sequence of zero or more rewrite rules from Fig. 4, where a congruence can be applied at any point. The congruence ensures that *par* and *times* are commutative operators. We are particularly interested in special derivations, called proofs.

Definition 1. *A proof in BV is a derivation $P \longrightarrow \mathrm{I}$ from a predicate P to the unit I. When such a derivation exists, we say that P is provable, and write $\vdash P$.*

$$
\begin{array}{lll}
P ::= & \mathrm{I} & \text{(unit)} \\
 & \alpha & \text{(atom)} \\
 & \overline{\alpha} & \text{(co-atom)} \\
 & P \parr P & \text{(par)} \\
 & P \otimes P & \text{(times)} \\
 & P \mathbin{;} P & \text{(seq)}
\end{array}
\qquad
\begin{array}{ll}
C\{\, \overline{\alpha} \parr \alpha \,\} \longrightarrow C\{\, \mathrm{I} \,\} & \text{(atomic interaction)} \\
C\{\, (P \otimes Q) \parr R \,\} \longrightarrow C\{\, P \otimes (Q \parr R) \,\} & \text{(switch)} \\
C\{\, (P \mathbin{;} Q) \parr (R \mathbin{;} S) \,\} \longrightarrow C\{\, (P \parr R) \mathbin{;} (Q \parr S) \,\} & \text{(sequence)}
\end{array}
$$

$(P, \parr, \mathrm{I})$ and $(P, \otimes, \mathrm{I})$ are commutative monoids, and $(P, ;, \mathrm{I})$ is a monoid.

Figure 1: The syntax and semantics of BV

$$
\begin{array}{lll}
P ::= & 1 & \text{(unit)} \\
 & a.P & \text{(atom)} \\
 & \overline{a}.P & \text{(co-atom)} \\
 & P \parr P & \text{(par)} \\
A ::= & \tau \mid a \mid \overline{a} & \text{(action)}
\end{array}
$$

$$
\frac{}{a.P \xrightarrow{a} P} \qquad \frac{}{\overline{a}.P \xrightarrow{\overline{a}} P}
$$

$$
\frac{P \xrightarrow{A} Q}{P \parr R \xrightarrow{A} Q \parr R} \qquad \frac{P \xrightarrow{\overline{a}} P' \qquad Q \xrightarrow{a} Q'}{P \parr Q \xrightarrow{\tau} P' \parr Q'}
$$

Figure 2: The syntax and labelled transition semantics of BCCS, plus the symmetric rules for parallel composition.

To explore the theory of proofs, linear negation is defined as a function over propositions. Notice in the syntax in Fig. 3 linear negation applies only to atoms.

Definition 2. *Linear negation is defined by the following function from predicates to predicates.*

$$\overline{\overline{\alpha}} = \alpha \qquad \overline{P \otimes Q} = \overline{P} \parr \overline{Q} \qquad \overline{P \parr Q} = \overline{P} \otimes \overline{Q} \qquad \overline{\mathrm{I}} = \mathrm{I} \qquad \overline{P \mathbin{;} Q} = \overline{P} \mathbin{;} \overline{Q}$$

Linear negation defines de Morgan dualities. As in linear logic, the multiplicatives $\otimes$ and $\parr$ are de Morgan dual. Sequential composition and the unit are self-dual, in the sense that their de Morgan dual operators are themselves. A generalisation of cut elimination in BV can be formulated as follows.

Theorem 1 (Guglielmi 2007)**.** *If* $\vdash C\{\, P \otimes \overline{P} \,\}$ *then* $\vdash C\{\, \mathrm{I} \,\}$.

The above formulation of cut elimination may look exotic, however, its consequences including the consistency of BV are as expected for a proof calculus. Another notable corollary of cut elimination is that *linear implication*, written $P \multimap Q$ and defined as $\overline{P} \parr Q$, defines a precongruence.

2.2 Embedding Basic CCS in BV

For a self-contained presentation, and comparison to the methodology of the body of the paper, we recall established results [6] for a fragment of CCS. The fragment of CCS, referred to as BCCS, consists of complementary atoms a and $\overline{a}$, prefix and parallel composition. The embedding of BCCS processes as propositions in BV is defined such that:

$$[\![1]\!]_{\mathsf{B}} = \mathrm{I} \qquad [\![P \parr Q]\!]_{\mathsf{B}} = [\![P]\!]_{\mathsf{B}} \parr [\![Q]\!]_{\mathsf{B}} \qquad [\![a.P]\!]_{\mathsf{B}} = a \mathbin{;} [\![P]\!]_{\mathsf{B}} \qquad [\![\overline{a}.P]\!]_{\mathsf{B}} = \overline{a} \mathbin{;} [\![P]\!]_{\mathsf{B}}$$

The labelled transitions system in Fig. 2 can be used to define a completed trace semantics.

Definition 3. *Completed traces are defined inductively as follows. The empty completed trace is defined using the unit* I*. If* T *is a completed trace then* $a \mathbin{;} T$ *is a completed trace and* $\overline{a} \mathbin{;} T$ *is a completed trace.*

Successful termination is indicated by the $\checkmark$ *predicate defined such that* $1\checkmark$ *holds and if* $P\checkmark$ *and* $Q\checkmark$ *then* $(P \parr Q)\checkmark$.

Completed traces are associated with processes such that if $P\checkmark$ then P has completed trace I*; and, inductively if Q has completed trace T then the following hold:*

- *If* $P \xrightarrow{\tau} Q$ *then P has completed trace T.*
- *If* $P \xrightarrow{a} Q$ *then P has completed trace $\overline{a}\,;T$.*
- *If* $P \xrightarrow{\overline{a}} Q$ *then P has completed trace $a\,;T$.*

By employing a procedure for extracting successful executions form proofs, the following relationship between completed traces and proofs can be established.

Theorem 2 (Bruscoli 2002)**.** *If T is a completed trace and P is a process, then $\vdash [\![P]\!]_B \parallel T$ if and only if P has completed trace T.*

Combined with cut elimination (Theorem 1), the above results can be used to establish the soundness of linear implication with respect to completed trace inclusion.

Corollary 1. *For BCCS processes P and Q, if $\vdash [\![P]\!]_B \multimap [\![Q]\!]_B$ then, for all completed traces T, if P has completed trace T then Q has completed trace T.*

The above result is based on a decomposition result for structures relating proofs in BV to a *lazy* variant of BV. In lazy BV, *atomic interaction* can only be applied inside contexts, called *left contexts*, of the form

$$\mathcal{L}\{\ \cdot\ \} ::= \{\ \cdot\ \} \mid \mathcal{L}\{\ \cdot\ \}\,;P \mid \mathcal{L}\{\ \cdot\ \} \parallel P \mid \mathcal{L}\{\ \cdot\ \} \otimes P.$$

For example, in a lazy proof of the structure $(a\,;(b \parallel c)) \parallel \left(\overline{a} \parallel \overline{b}\right)\,;\overline{c}$, the *atomic interaction* rule must apply first to a before b, and finally c. A established property of lazy BV is that any proof of a proposition in BV in which *times* $\otimes$ never appears can be renormalised into a proof in the lazy variant of BV.

Proposition 1. *If P contains no times operator and $\vdash P$ then there exists Q such that $P \longrightarrow Q$ using the sequence rule and $\vdash Q$ using only the atomic interaction rule applied in left contexts.*

Bruscoli [6] hypothesised that the restriction forbidding times from appearing in structures could be lifted. Here we show it is impossible to fully lift that restriction on the *times* operator. To see this, observe that the following structure has a proof in BV.

$$\vdash \left(\left(\left(\overline{a}\,;\overline{b}\right) \otimes \left(\overline{d}\,;\overline{e}\right)\right)\,;\left(\overline{c} \otimes \overline{f}\right)\right) \parallel (a\,;((b\,;c) \otimes (\overline{g} \parallel g))) \parallel \left(d\,;\left((e\,;f) \otimes \left(\overline{h} \parallel h\right)\right)\right)$$

The above structure has no proof in the lazy variant of BV, where interaction is restricted to left contexts. The difficulty is that the atoms g and h must interact before the sequence rule can be correctly applied. Fortunately, we observe that it is possible to relax the restriction forbidding the *times* operator for propositions of certain forms, as expressed by the following proposition.

Proposition 2. *Assume P and Q are propositions in which no times operator occurs. If $\vdash P \multimap Q$, the there exists R such that $P \multimap Q \longrightarrow R$ using only the sequence and switch rule, and $\vdash R$ using only the atomic interaction rule applied in a lazy context.*

The proof works by combining the *splitting* technique used in the proof of cut elimination, with the strategy of permuting interaction rules as in Lemma 1. Observe that embeddings of BCCS processes do not use the times operator. Hence, by the above observation, for any BCCS processes P and Q, if $\vdash [\![P]\!]_{\mathsf{B}} \multimap [\![Q]\!]_{\mathsf{B}}$ then we can always construct a lazy proof of the same proposition. For example, $[\![b \parallel c]\!]_{\mathsf{B}} \multimap \left[\!\!\left[\overline{a}.b.\overline{d} \parallel a.c.d\right]\!\!\right]_{\mathsf{B}}$ has the following lazy proof.

$$\begin{aligned}\left(\overline{b} \otimes \overline{c}\right) \parallel \left(\overline{a}\,;b\,;\overline{d}\right) \parallel (a\,;c\,;d) \longrightarrow\ & (\overline{a} \parallel a)\,;\left(\left(\overline{b} \parallel b\right) \otimes (\overline{c} \parallel c)\right)\,;\left(\overline{d} \parallel d\right)\\ \longrightarrow\ & \left(\left(\overline{b} \parallel b\right) \otimes (\overline{c} \parallel c)\right)\,;\left(\overline{d} \parallel d\right) \longrightarrow (\overline{c} \parallel c)\,;\left(\overline{d} \parallel d\right) \longrightarrow \left(\overline{d} \parallel d\right) \longrightarrow \mathrm{I}\end{aligned}$$

The above lazy derivation suggests a truly concurrent (non-interleaving) execution for process $\overline{a}.b.\overline{d} \parallel a.c.d$: Firstly, $\overline{a}$ and a interact; secondly, the two events b and c happen truly concurrently (indicated by *times*, intuitively indicating events occur within overlapping time frames); thirdly $\overline{d}$ and d interact. Indeed we can obtain a tighter truly concurrent process semantics for linear implication, for the fragment of BV without *times*. In related work [7], an extension of series-parallel pomsets [9] was introduced. The extension uses graph homomorphisms, called *interacting* homomorphisms, that allow causal dependencies to be strengthened and then concurrent complementary atoms to cancel each other out (ignoring the application specific "artefacts"). In that related work [7], series-parallel processes, which correspond to the fragment of BV without *times*, were shown to be sound and complete with respect to the axioms of BV without *times* (the key technical device is interpolation [7]Lemma 5.11). We mention this result to show that, although Corollary 1 establishes that linear implication is sound with respect to completed trace inclusion, a finer sound and complete semantics takes into account the subtleties of pomsets. A related observation is that proofs themselves are truly concurrent [15].

3 First-order System BV1 and Mobile Processes

The proof calculus BV has been extended with additives [12], exponentials [11, 25] and first-order quantifiers [13]. The first-order extension features a novel de Morgan dual pair of nominal quantifiers "new" and "wen" that do not have a counterpart in linear logic. These operators were introduced to model private names as in variants of the π-calculus. In this section we clarify this story in terms of a first-order extension of BV, called BV1. We highlight the missing link for establishing the soundness of linear implication with respect to completed trace inclusion. Thereby we enable the soundness results for linear implication in BCCS to be extended to the π-calculus.

$$
\begin{array}{rll}
P ::= & \alpha & \text{(atom)} \\
 & \overline{\alpha} & \text{(co-atom)} \\
 & 1 & \text{(unit)} \\
 & \forall x.P & \text{(all)} \\
 & \exists x.P & \text{(some)} \\
 & \mathit{И}x.P & \text{(new)} \\
 & \mathit{Э}x.P & \text{(wen)} \\
 & P \parallel P & \text{(par)} \\
 & P \otimes P & \text{(times)} \\
 & P \,;\, P & \text{(seq)}
\end{array}
$$

Figure 3: BV1 syntax.

3.1 The Syntax and Semantics of BV1

The proof calculus BV1 extends BV with the first-order quantifiers *forall*, *new*, *wen* and *exists*. The syntax is presented in Fig. 3. The rewrite rules, in Fig 4, can be applied in any context, where a context is a predicate with a hole of the form $C\{\ \} ::= \{\ \cdot\ \} \mid C\{\ \} \odot P \mid P \odot C\{\ \} \mid \mathcal{O}x.C\{\ \}$, where $\odot \in \{;, \parallel, \otimes\}$ and $\mathcal{O} \in \{\exists, \forall, \mathit{И}, \mathit{Э}\}$. Note the direction of rewriting is opposite to the direction of linear implication.

The rewrite rules are defined modulo a congruence. The congruence extends the congruence for BV with α-conversion and *equivariance* for nominal quantifiers.

$$\mathit{И}x.\mathit{И}y.P \equiv \mathit{И}y.\mathit{И}x.P \qquad \mathit{Э}x.\mathit{Э}y.P \equiv \mathit{Э}y.\mathit{Э}x.P \quad \text{(equivariance)}$$

Equivariance allows both nested *new* and *wen* operators to be exchanged. Note that for *exists* and *forall* equivariance is a derived property; but must be explicitly induced for nominals.

Freshness is defined such that x is fresh for a predicate P, written $x \# P$, if and only if x is not a member of the set of free variables of P, such that all quantifiers bind variables in their scope. As standard, substitution is assumed to avoid the capture of free variables.

Considerable creativity is permitted when defining terms that form the atoms of the calculus. In work on session types [8], atoms ranged over tuples containing any datatype, with respect a subtype preorder. For the π-calculus embeddings in this paper, atoms are simply pairs of first order variables representing

$$C\{\ \overline{\alpha} \parallel \alpha\ \} \longrightarrow C\{\ \mathrm{I}\ \} \quad \text{(atomic interaction)} \qquad C\{\ P \parallel (Q \otimes S)\ \} \longrightarrow C\{\ (P \parallel Q) \otimes S\ \} \quad \text{(switch)}$$

$$C\{\ (P\ ;\ Q) \parallel (R\ ;\ S)\ \} \longrightarrow C\{\ (P \parallel R)\ ;\ (Q \parallel S)\ \} \quad \text{(sequence)}$$

$$C\{\ \forall x.P \parallel R\ \} \longrightarrow C\{\ \forall x.(P \parallel R)\ \} \text{ only if } x \# R \quad \text{(extrude1)} \qquad C\{\ \forall x.\mathrm{I}\ \} \longrightarrow C\{\ \mathrm{I}\ \} \quad \text{(tidy1)}$$

$$C\{\ \forall x.(P\ ;\ S)\ \} \longrightarrow C\{\ \forall x.P\ ;\ \forall x.S\ \} \quad \text{(medial1)} \qquad C\{\ \exists x.P\ \} \longrightarrow C\{\ P\{^{v}/_{x}\}\ \} \quad \text{(select1)}$$

$$C\{\ И x.P \parallel Э x.Q\ \} \longrightarrow C\{\ И x.(P \parallel Q)\ \} \quad \text{(close)} \qquad C\{\ И x.\mathrm{I}\ \} \longrightarrow C\{\ \mathrm{I}\ \} \quad \text{(tidy name)}$$

$$C\{\ И x.P \parallel R\ \} \longrightarrow C\{\ И x.(P \parallel R)\ \} \text{ only if } x \# R \quad \text{(extrude new)}$$

$$C\{\ Э x.P\ \} \longrightarrow C\{\ И x.P\ \} \quad \text{(fresh)} \qquad C\{\ И x.Э y.P\ \} \longrightarrow C\{\ Э y.И x.P\ \} \quad \text{(new wen)}$$

$$C\{\ И x.(P\ ;\ S)\ \} \longrightarrow C\{\ И x.P\ ;\ И x.S\ \} \quad \text{(medial new)}$$

$$C\{\ Э x.P \odot Э x.S\ \} \longrightarrow C\{\ Э x.(P \odot S)\ \} \text{ where } \odot \in \{\parallel, ;\} \quad \text{(suspend name)}$$

$$C\{\ Э x.P \odot R\ \} \longrightarrow C\{\ Э x.(P \odot R)\ \} \text{ where } \odot \in \{\parallel, ;\} \text{ only if } x \# R \quad \text{(left wen)}$$

$$C\{\ R \odot Э x.Q\ \} \longrightarrow C\{\ Э x.(R \odot Q)\ \} \text{ where } \odot \in \{\parallel, ;\} \text{ only if } x \# R \quad \text{(right wen)}$$

$$C\{\ \forall x.\mathcal{O} y.P\ \} \longrightarrow C\{\ \mathcal{O} y.\forall x.P\ \} \text{ for } \mathcal{O} \in \{И, Э\} \quad \text{(all name)}$$

Figure 4: Rewrite rules for predicates in system BV1.

a channel and value passed on the given channel. Value passing extensions of the π-calculus could be considered by extending with terms constructed from function symbols, constants and first-order variables. However, an expressive term language can increase considerably the complexity of proof search [16].

A derivation is a sequence of zero or more rewrite rules from Fig. 4, where the structural congruence can be applied at any point. We are particularly interested in derivations that constitute proofs.

Definition 4. *A proof in BV1 is a derivation $P \longrightarrow \mathrm{I}$ from a predicate P to the unit I. When such a derivation exists, we say that P is provable, and write $\vdash P$.*

As a technical device we define a form of restricted context called a killing context, consisting of a hole surrounded by quantifiers $\forall$ and И.

Definition 5 (killing context)**.** *A killing context $\mathcal{T}\{\ \}$ is defined by the following grammar, where $\{\cdot\}$ is a hole into which any process can be plugged: $\mathcal{T}\{\ \} ::= \{\cdot\} \mid \forall x.\mathcal{T}\{\ \} \mid И x.\mathcal{T}\{\ \}$.*

Splitting, the main technical lemma required for proving cut elimination is formulated as follows for BV1. In the sequent calculus, any connective can be selected and a corresponding rule applied. Splitting renormalises proofs to generalise this feature of sequents to the calculus of structures.

Lemma 1 (Splitting)**.** *The following statements hold.*

- *For any atom α, if $\vdash \overline{\alpha} \parallel Q$, then there exist killing context $\mathcal{T}\{\ \}$ such that $Q \longrightarrow \mathcal{T}\{\ \alpha\ \}$, where if x appears in $\mathcal{T}\{\ \}$ then $x \# \alpha$.*

- *For any atom α, if $\vdash \alpha \parallel Q$, then there exist killing context $\mathcal{T}\{\ \}$ such that $Q \longrightarrow \mathcal{T}\{\ \overline{\alpha}\ \}$, where if x appears in $\mathcal{T}\{\ \}$ then $x \# \alpha$.*
- *If $\vdash (P \otimes Q) \parallel R$, then there exist predicates V and W such that $\vdash P \parallel V$ and $\vdash Q \parallel W$, and killing context $\mathcal{T}\{\ \}$ such that $R \longrightarrow \mathcal{T}\{\ V \parallel W\ \}$ and if x appears in $\mathcal{T}\{\ \}$ then $x \# (P \otimes Q)$.*
- *If $\vdash (P\ ;\ Q) \parallel R$, then there exist predicates V and W such that $\vdash P \parallel V$ and $\vdash Q \parallel W$, and killing context $\mathcal{T}\{\ \}$ such that $R \longrightarrow \mathcal{T}\{\ V\ ;\ W\ \}$ and if x appears in $\mathcal{T}\{\ \}$ then $x \# (P\ ;\ Q)$.*
- *If $\vdash \text{И}x.P \parallel Q$, then there exist predicates V and W where $x \# V$ and $\vdash P \parallel W$ and either $V = W$ or $V = \Theta x.W$, such that derivation $Q \longrightarrow V$ holds.*
- *If $\vdash \Theta x.P \parallel Q$, then there exist predicates V and W where $x \# V$ and $\vdash P \parallel W$ and either $V = W$ or $V = \text{И}x.W$, such that derivation $Q \longrightarrow V$ holds.*
- *If $\vdash \exists x.P \parallel Q$, then there exist predicate V and value v such that $\vdash P\{{}^{v}/_{x}\} \parallel V$, and killing context $\mathcal{T}\{\ \}$ such that $Q \longrightarrow \mathcal{T}\{\ V\ \}$ and if y appears in $\mathcal{T}\{\ \}$ then $y \# (\exists x.P)$.*
- *If $\vdash \forall x.P \parallel Q$ then, for any term t, $\vdash P\{{}^{t}/_{x}\} \parallel Q$.*

Linear negation extends the de Morgan dualities for BV to quantifiers. The first-order quantifiers $\exists$ and $\forall$ are de Morgan dual, as are the nominal quantifiers И and Θ.

Definition 6. *Linear negation is defined by the following function from predicates to predicates.*

$$\overline{\overline{\alpha}} = \alpha \qquad \overline{P \otimes Q} = \overline{P} \parallel \overline{Q} \qquad \overline{P \parallel Q} = \overline{P} \otimes \overline{Q} \qquad \bar{1} = 1 \qquad \overline{P\ ;\ Q} = \overline{P}\ ;\ \overline{Q}$$

$$\overline{\forall x.P} = \exists x.\overline{P} \qquad \overline{\exists x.P} = \forall x.\overline{P} \qquad \overline{\text{И}x.P} = \Theta x.\overline{P} \qquad \overline{\Theta x.P} = \text{И}x.\overline{P}$$

Cut elimination holds for BV1 as a consequence of cut-elimination for MAV1 [13]. The key ingredients are splitting (Lemma 1), and a context lemma that extends splitting to arbitrary contexts.

Theorem 3. *If $\vdash C\{\ P \otimes \overline{P}\ \}$ then $\vdash C\{\ 1\ \}$.*

Consequences of cut elimination include that linear implication defines a precongruence. Consistency can be established immediately in the sense that for any provable predicate with at least one atom, the linear negation is not provable. Cut elimination is of course the corner stone of a proof calculus.

3.2 Embedding π-calculus Processes as Predicates in BV1

We embed a fragment of the π-calculus in BV1 presented in Fig. 5. The fragment features parallel composition, input and output actions and private name binders.

$$[\![1]\!]_\pi = 1 \qquad [\![P \parallel Q]\!]_\pi = [\![P]\!]_\pi \parallel [\![Q]\!]_\pi \qquad [\![\nu x.P]\!]_\pi = \text{И}x.[\![P]\!]_\pi$$

$$[\![x(z).P]\!]_\pi = \exists z.(xz\ ;\ [\![P]\!]_\pi) \qquad [\![\overline{x}z.P]\!]_\pi = \overline{xz}\ ;\ [\![P]\!]_\pi$$

Note that the constant 1 is used to indicate the successfully terminated process. We use the symbol 1, rather than 0 frequently used in the literature, so as to reserve 0 for representing deadlock.

We assume the reader is familiar with the labelled transition system for the π-calculus. A small but notable syntactic departure from the literature is the use of action $\overline{x}[z]$ to represent a bound output. This syntax is chosen to disambiguate semantically distinct concepts. In this paper we would like to discuss both the *input* of the π-calculus, and, later in the paper, the *private input* of the πI-calculus [23]. Traditionally, these inputs use the same syntax but have distinct semantics. To disambiguate which input we are discussing we use $x(z)$ for the π-calculus input, which can receive both private and public

$$P ::= 1 \quad \text{(success)} \qquad \frac{}{x(y).P \xrightarrow{x(y)} P} \qquad \frac{}{\overline{x}y.P \xrightarrow{\overline{x}y} P} \qquad \frac{P \xrightarrow{\overline{x}z} Q}{\nu z.P \xrightarrow{\overline{x}[z]} Q}\ x \neq z$$

$$\nu x.P \quad \text{(nu)}$$

$$x(y).P \quad \text{(input)}$$

$$\overline{x}y.P \quad \text{(output)}$$

$$P \parallel P \quad \text{(par)} \qquad \frac{P \xrightarrow{A} Q}{\nu x.P \xrightarrow{A} \nu x.Q}\ x \notin \mathrm{n}(A) \qquad \frac{P \xrightarrow{A} Q}{P \parallel R \xrightarrow{A} Q \parallel R}\ \begin{array}{l}\text{if } A = \overline{x}[z] \\ \text{or } A = x(z) \\ \text{then } z \# R\end{array}$$

$$A ::= \tau \mid \overline{x}[z] \mid \overline{x}z \mid x(z) \quad \text{(actions)} \qquad \frac{P \xrightarrow{\overline{x}[z]} P' \quad Q \xrightarrow{x(z)} Q'}{P \parallel Q \xrightarrow{\tau} \nu z.(P' \parallel Q')} \qquad \frac{P \xrightarrow{\overline{x}y} P' \quad Q \xrightarrow{x(z)} Q'}{P \parallel Q \xrightarrow{\tau} P' \parallel Q'\{^{y}/_{z}\}}$$

Figure 5: Syntax and labelled transition system for π-calculus, plus the symmetric rules for parallel composition. Function $n(.)$ is such that $n(x(y)) = n(\overline{x}[y]) = n(\overline{x}y) = \{x, y\}$ and $n(\tau) = \emptyset$; and $x \# P$ is such that x is fresh for P, where $z(x).P$ and $\nu x.P$ bind x in P.

names, and $x[z]$ for πI-calculus input, which can receive a name if it is guaranteed to be private. The natural dual to private input is the output of a fresh private name, hence we prefer the syntax $\overline{x}[z]$ even for the action outputting a private name in the π-calculus. In the logical embedding, the intuitive duality between $x[z]$ and $\overline{x}[z]$ is put on a precise foundation by the use of dual operators $\mathit{И}$ and $\Im$ when their semantics are presented in BV1.

Well formed completed traces can be defined independently of processes and associated with processes according to the following definitions.

Definition 7. *Define completed traces inductively as follows. The unit* I *is a completed trace and if* T *is a completed trace then:*

- $\overline{xy}\,;T$ *is a completed trace.*
- $xy\,;T$ *is a completed trace.*
- $\Im y.(xy\,;T)$ *is a completed trace, where* $x \neq y$.

The checkmark $\checkmark$ *predicate holds for processes that have already successfully terminated.* $1\checkmark$ *holds and, inductively, if* $P\checkmark$ *and* $Q\checkmark$ *then:* $(P \parallel Q)\checkmark$ *holds and* $(\nu x.P)\checkmark$ *holds. Completed traces can be associated with processes such that if* $P\checkmark$ *holds then* P *has the completed trace* I*, and, inductively, if* Q *has completed trace* T *then the following hold:*

- *If* $P \xrightarrow{\overline{x}[z]} Q$ *then* P *has completed trace* $\Im z.(xz\,;T)$.
- *If* $P \xrightarrow{\overline{x}z} Q$ *then* P *has completed trace* $xz\,;T$.
- *If* $P \xrightarrow{\tau} Q$ *then* P *has completed trace* T.

And also, if $P \xrightarrow{x(z)} Q$ *and* $Q\{^{y}/_{x}\}$ *has completed trace* T *then* P *has completed trace* $\overline{xy}\,;T$.

Stylistically, a late transition system is presented in Fig. 5. Observe that when an input and free output interact, a substitution is applied to unify the input variable and free output. This is taken into account also in the definition of traces above by considering all possible substitutions for the input variable z in the continuation y. For example, the process $a(x).(\overline{x}x \parallel a(y))$ has completed trace $\overline{aa}$. This completed trace corresponds to the following two transitions in the labelled transition semantics, where a substitution for x enabling the τ transition is applied to both the process and the input transition.

$$a(x).(\overline{x}x \parallel a(y)) \xrightarrow{a(x)} \overline{x}x \parallel a(y) \qquad (\overline{x}x \parallel a(y))\{^{a}/_{x}\} = \overline{a}a \parallel a(y) \xrightarrow{\tau} \mathrm{I}$$

In the above example $\vdash \overline{aa} \parallel [\![a(x).(\overline{x}x \parallel a(y))]\!]_\pi$ formally relating completed trace $\overline{aa}$ with the given process using the semantics of BV1. In general, we can always obtain such a proof in BV1 correspond to a completed trace. The following proposition verifies this is always the case, and follows by cut elimination (Theorem 3), and induction over the structure of derivations in the labelled transition system.

Proposition 3. *If T is a completed trace and P is a π-calculus process, then if P has completed trace T, then $\vdash T \parallel [\![P]\!]_\pi$.*

The main technical contribution of this paper is to prove the following converse proposition. The proposition establishes that whenever a completed trace and process has a proof, we can always construct a sequence of labelled transitions corresponding to the completed trace. Notice that we only assume that T is a well formed trace, in the following and do not assume a priori that T is associated with a completed trace of process P.

Proposition 4. *If T is a completed π-calculus trace and P a π-calculus process, then if $\vdash T \parallel [\![P]\!]_\pi$ then P has completed trace T.*

We postpone the proof of the above proposition until the necessary techniques are developed in the next section. The techniques developed for the above proposition form the main technical contribution of this paper. To understand why the above proposition is important, observe that it is the missing step in the following theorem. By "missing step", we mean that the result is partly established by cut elimination for BV1 which was established in a previous paper [13].

Theorem 4 (completed trace inclusion)**.** *For π-calculus processes P and Q, if $\vdash [\![P]\!]_\pi \multimap [\![Q]\!]_\pi$ then, for all completed traces T, if P has completed trace T then P also has completed trace T.*

Proof. Assume $\vdash [\![P]\!]_\pi \multimap [\![Q]\!]_\pi$ holds and P has completed trace T. By Proposition 3, $\vdash T \parallel [\![P]\!]_\pi$ holds. Hence by Theorem 3, $\vdash T \parallel [\![Q]\!]_\pi$. Thereby by Proposition 4, Q has completed trace T. □

Since completed trace inclusion is amongst the coarsest preorders over processes, the above result is a minimal correctness condition for a preorder over processes. We emphasise that, as with BV linear implication is much finer than trace inclusion, respecting causality as in pomsets. We leave a sound and complete extension of series-parallel pomsets to BV1 without *times*, as future work. The tightest non-interleaving semantics we are currently aware of are based on extensions of event structures [28].

4 Decompositions and the Extraction of Successful Executions

This section is the main technical contribution of the paper. In this section, we extend decomposition properties for BV to BV1. The decomposition properties control the interaction of atoms, extrusion of quantifiers and instantiation of existential quantifiers. Along with splitting, these decomposition properties enable proofs to be renormalised such that labelled transitions can be extracted.

4.1 Decomposition for the Atoms and Lazy BV1

To define a lazy variant of BV1, where rules are applied in restricted contexts, we require a notion of a left context in which interactions are permitted. Left contexts are special contexts defined such that the hole only appears on the left of a "seq". In terms of processes, this means that anything inside the context is ready to happen next, since there are no events that must happen before the events in the context.

Definition 8. *A left context $\mathcal{L}\{\ \cdot\ \}$ is defined according to the following grammar:*

$$\mathcal{L}\{\ \cdot\ \} ::= \{\ \cdot\ \} \mid \mathcal{L}\{\ \cdot\ \}\ ;\ P \mid \mathcal{L}\{\ \cdot\ \} \parallel P \mid \mathcal{L}\{\ \cdot\ \} \otimes P \mid И x.\mathcal{L}\{\ \cdot\ \} \mid \forall x.\mathcal{L}\{\ \cdot\ \}$$

The decomposition properties for atoms in BV [6] extend to BV1: atomic interactions can be pushed upwards in a proof until they are in a left context. In BV1, we should also permute up the tidy rules, since they may obstruct an interaction from being pushed up. To see this consider the following proof:

$$\begin{aligned}
\left(\overline{ab}\ ;\ \nu z.\overline{cz}\right) \parallel \exists x.(ax\ ;\ \exists y.cy) &\longrightarrow \left(\overline{ab}\ ;\ \nu z.\overline{cz}\right) \parallel (ab\ ;\ \exists y.cy)\\
&\longrightarrow \left(\overline{ab} \parallel ab\right)\ ;\ (\nu z.\overline{cz} \parallel \exists y.cy)\\
&\longrightarrow \left(\overline{ab} \parallel ab\right)\ ;\ \nu z.(\overline{cz} \parallel \exists y.cy)\\
&\longrightarrow \left(\overline{ab} \parallel ab\right)\ ;\ \nu z.(\overline{cz} \parallel cz) \longrightarrow \left(\overline{ab} \parallel ab\right)\ ;\ \nu z.\mathrm{I} \longrightarrow \left(\overline{ab} \parallel ab\right) \longrightarrow \mathrm{I}
\end{aligned}$$

In the above the the tidy name rule followed by one of the atomic interaction rules can be permuted upwards until they are in a left context. The last line of the above proof is thereby transformed to a proof where rules are only applied in left contexts as follows.

$$\left(\overline{ab} \parallel ab\right)\ ;\ \nu z.(\overline{cz} \parallel cz) \longrightarrow \nu z.(\overline{cz} \parallel cz) \longrightarrow \nu z.\mathrm{I} \longrightarrow \mathrm{I}$$

Based on this observation we define the interaction fragment of BV1. A lazy proof restricts the context in which rules of the interaction fragment can be applied.

Definition 9. *The interaction fragment of BV1 consists of the rules atomic interaction, tidy1 and tidy name. The core fragment consists of all other rules of BV1. A lazy proof in BV1 is a proof where the rules of the interaction fragment are only applied in a left context.*

There are few subtleties, which we explain before presenting the decomposition. A restriction is that, as in BV, the interaction fragment does not permute over all rules. We should take care about how the *times* operator appears in derivations. To see this, consider the following derivation involving *tidy name*, *medial new* and the *times* operator.

$$C\{\ И x.(P\ ;\ (И y.\mathrm{I} \otimes (Q\ ;\ R))\ ;\ S)\ \} \longrightarrow C\{\ И x.(P\ ;\ Q\ ;\ R\ ;\ S)\ \} \longrightarrow C\{\ И x.(P\ ;\ Q)\ ;\ И x.(R\ ;\ S)\ \}$$

The *tidy name* rule in the above derivation cannot be permuted above the *medial new* rule in BV1. It is possible to extend BV1 with the *co-sequence* and *co-medial new* rules below.

$$C\{\ (P \otimes Q)\ ;\ (R \otimes S)\ \} \longrightarrow C\{\ (P\ ;\ R) \otimes (Q\ ;\ S)\ \} \quad \text{co-sequence}$$

$$C\{\ И x.(P \otimes Q)\ \} \longrightarrow C\{\ И x.P \otimes И x.Q\ \} \quad \text{co-medial name}$$

The above *co-medial* rules would allow the *tidy name* rule to be permuted upwards as follows:

$$\begin{aligned}
C\{\ И x.(P\ ;\ (И y.\mathrm{I} \otimes (Q\ ;\ R))\ ;\ S)\ \} &\longrightarrow C\{\ И x.((И y.\mathrm{I} \otimes (P\ ;\ Q\ ;\ R))\ ;\ S)\ \}\\
&\longrightarrow C\{\ И x.(И y.\mathrm{I} \otimes (P\ ;\ Q\ ;\ R\ ;\ S))\ \}\\
&\longrightarrow C\{\ И x.И y.\mathrm{I} \otimes И x.(P\ ;\ Q\ ;\ R\ ;\ S)\ \}\\
&\longrightarrow C\{\ И x.\mathrm{I} \otimes И x.(P\ ;\ Q\ ;\ R\ ;\ S)\ \} \longrightarrow C\{\ И x.(P\ ;\ Q\ ;\ R\ ;\ S)\ \}
\end{aligned}$$

Although the above derivation introduces extra tidy new rules, the distance to the top of the derivation strictly decreases. Hence a multiset-based measure can be defined that accounts for the distance of interaction rules from the top of a proof, as in related work [25]. A similar scenario holds for universal quantifiers, requiring the following *co-medial1* rule.

$$C\{\ \forall x.(P \otimes Q)\ \} \longrightarrow C\{\ \forall x.P \otimes \forall x.Q\ \} \quad \text{co-medial1}$$

This leads us to a first attempt at a decomposition property.

Proposition 5. *If $P \longrightarrow R$ in BV1 then there exists Q such that:*

- *$P \longrightarrow Q$ in the core fragment of BV1 plus the rules co-sequence, co-medial1 and co-medial name.*
- *$Q \longrightarrow R$ in the interaction fragment of BV1.*

Fortunately, the embedding of processes as predicates in BV1 that we employ does not require the *times* operator. For predicates in BV1 without *times* the above decomposition can be tightened as follows, by removing the need for additional *co-medial* rules.

Proposition 6. *If $P \longrightarrow R$ in BV1, where P and R contain no times operator, then there exists Q such that:*

- *$P \longrightarrow Q$ in the core fragment of BV1.*
- *$Q \longrightarrow R$ in the interaction fragment of BV1.*

The proof is along the lines for BV [6], where we permute up the topmost interaction rule that is not applied in a left context. Notice *sequence* rules may be introduced as rules are permuted upwards, and the use of *times* operators are not restricted.

Lemma 2. *If $P \longrightarrow Q$ is a rule in the interaction fragment and Q has a lazy proof using the interaction fragment and sequence rule, then P has a lazy proof using the interaction fragment and sequence rule.*

The above properties together enable us to establish that for predicates without the *times* operator, any proof can be transformed to a lazy proof.

Proposition 7. *For predicates P without the times operator, if $\vdash P$ in BV1 then there is a lazy proof of P in BV1.*

Note that, in a derivation where the *times* operator never appears in the conclusion, there is only one way in which a *times* operator can be introduced, that cannot be removed by the unit identities: there must be a *switch* rule of the form $(P \otimes 1) \parallel Q \longrightarrow P \otimes (1 \parallel Q)$. All such switch instances can be removed from a derivation.

Although we cannot construct a lazy proof for all predicates containing times, we can handle a large class of predicates. The class we will be concerned with is predicates of the form $P \parallel Q$, where P contains no *par* operator, but possibly *times*, and Q contains no *times*, but possibly *par*. Fortunately, this class of predicates are exactly those required for comparing embeddings of processes. By combining the above decomposition properties with splitting we obtain the following more general property transforming proofs into lazy proofs.

Proposition 8. *If P and Q contain no times operator, then if $P \multimap Q$ has a proof in BV1, then $P \multimap Q$ also has a lazy proof in BV1.*

The above property is more general than required for traces. However, it highlights that lazy proofs exists for more general predicates. The more general form indicates that techniques for traces should extend to suitable pomsets [28, 21].

4.2 Eagerly Handling Scope Extrusion and Existential Quantifiers

By the lazy decomposition presented in the previous section, we can renormalise a proof into a lazy form and identify the bottommost rule in the interaction fragment. Since no rule introduces atoms or quantifiers, we can identify the atoms or quantifiers actively involved in the interaction rule in the conclusion of a proof. Since this rule is always in a left context and no rule below the interaction rule identified can remove atoms, we know that the atom or quantifier will appear in a left context in the conclusion of the proof.

To discuss processes on the conclusion we introduce process contexts.

Definition 10. *A process context $\mathcal{P}\{\ \cdot\ \}$ is defined according to the following grammar, where P is a π-calculus process:* $\mathcal{P}\{\ \cdot\ \} ::= \ \{\cdot\}\ \mid\ \mathcal{P}\{\cdot\} \parallel P\ \mid\ P \parallel \mathcal{P}\{\cdot\}\ \mid\ \nu x.\mathcal{P}\{\cdot\}$.

Observe that process contexts transform to left contexts when used in embeddings. When the bottommost rule in the interaction fragment is an instance of the *atomic interaction* rule, we identify six scenarios which we analyse in this section.

Scenario 1. The simplest scenario corresponds to the case where the next action of the process is the output of a free name on a channel. In the core fragment there is a derivation of the following form, where T is a completed trace, Q is a π-calculus process, $\mathcal{L}\{\ \}$ is a left context, $\mathcal{P}\{\ \}$ is a process context, and x and z are not bound by $\mathcal{P}\{\ \}$.

$$xz\,;T \parallel [\![\mathcal{P}\{\ \overline{x}z \parallel Q\ \}]\!]_\pi \longrightarrow \mathcal{L}\{\ xz \parallel \overline{xz}\ \}$$

Also, assume that $\mathcal{L}\{\ \mathrm{I}\ \}$ has a lazy proof. This scenario is no more complicated than for BCCS and BV. In this case, we can simply remove the instances of atoms xz and $\overline{xz}$ from the core fragment, yielding a proof of the form.

$$\vdash T \parallel [\![\mathcal{P}\{\ Q\ \}]\!]_\pi$$

Furthermore, it is easy to check, by structural induction on the process context, there is always the appropriate labelled transition outputting free name z on channel x, verified by the following proposition.

Proposition 9. *For π-calculus process Q and process context $\mathcal{P}\{\ \}$ that binds neither x nor z, There is a $\overline{x}z$ labelled transition from $\mathcal{P}\{\ \overline{x}z \parallel Q\ \}$ to $\mathcal{P}\{\ Q\ \}$.*

Observe that some rules become redundant so are removed from the proof in this process. For example consider a lazy proof of the following form.

$$\begin{array}{rl}
\Big(\underline{xz}\,;T\Big) \parallel \mathit{И}y.\Big(\underline{\overline{xz}}\,;P\Big) \longrightarrow & \Big(\underline{xz}\,;T\Big) \parallel \Big(\mathit{И}y.\underline{\overline{xz}}\,;\mathit{И}y.P\Big) \\
\longrightarrow & \Big(\underline{xz} \parallel \mathit{И}y.\underline{\overline{xz}}\Big)\,;(T \parallel \mathit{И}y.P) \\
\longrightarrow & \mathit{И}y.\Big(\underline{xz} \parallel \underline{\overline{xz}}\Big)\,;(T \parallel \mathit{И}y.P) \\
\longrightarrow & \mathit{И}y.\mathrm{I}\,;(T \parallel \mathit{И}y.P) \longrightarrow \mathrm{I}
\end{array}$$

In the above proof, after removing the atoms xz and $\overline{xz}$, instances of the rules *sequence*, *medial new*, *extrude new* are deleted. Also since $\mathit{И}y.\mathrm{I}\,;(T \parallel \mathit{И}y.P)$ has a lazy proof, the lowest rule in the interaction fragment is the instance of *tidy new* removing $\mathit{И}y$. Hence the redundant $\mathit{И}y.\mathrm{I}$ can also be removed yielding a lazy proof of $T \parallel \mathit{И}y.P$ as required.

Scenario 2. In the second scenario free output and input interact resulting in a τ transition. A technique is required to handle the existential quantifier in the embedding of the input. In the following assume T is a trace, P and Q are a π-calculus processes, $\mathcal{P}^i\{\ \}$ are process contexts and $\mathcal{L}\{\ \}$ is a left context. Consider when both atoms xz and $\overline{xz}$ can be traced to the embedding of a process, where there exists a derivation in the core fragment of BV1, where x and z are not bound by $\mathcal{P}^1\{\ \}$ and $\mathcal{P}^2\{\ \}$.

$$T \parallel \Big[\!\Big[\mathcal{P}^0\Big\{\ \mathcal{P}^1\{\ \overline{x}z.P\ \} \parallel \mathcal{P}^2\{\ x(y).Q\ \}\ \Big\}\Big]\!\Big]_\pi \longrightarrow \mathcal{L}\{\ \overline{xz} \parallel xz\ \}$$

Also, there exists a lazy proof of $\mathcal{L}\{\ \mathrm{I}\ \}$ in BV1.

In this scenario, observe that is always a τ transition where an input and unbounded output interact. The following proposition expresses the most general form of such a transition.

Proposition 10. *For any π-calculus process of the form $\mathcal{P}^0\{\ \mathcal{P}^1\{\ \overline{x}z.P\ \} \parallel \mathcal{P}^2\{\ x(y).Q\ \}\ \}$, where x and z are not bound by $\mathcal{P}^1\{\ \}$ and $\mathcal{P}^2\{\ \}$, there exists a τ transition to the process $\mathcal{P}^0\{\ \mathcal{P}^1\{\ P\ \} \parallel \mathcal{P}^2\{\ Q\{^z/_y\}\ \}\ \}$.*

The question is whether given the above lazy proof, can a proof of the following proposition be constructed.

$$T \parallel [\![\mathcal{P}^0\{\ \mathcal{P}^1\{\ P\ \} \parallel \mathcal{P}^2\{\ Q\{^z/_y\}\ \}\ \}]\!]_\pi$$

To do so, we require a technical decomposition result. For certain derivations, it is always possible to permute the instantiation of existential quantifiers to the bottom of the derivation, as long as it does not commute with an extrusion rule binding a variable in the term introduced. To understand this subtlety consider the following derivation, where $x \# P$.

$$И x.\overline{ax} \parallel \exists y.(ay; P) \longrightarrow И x.(\overline{ax} \parallel \exists y.(ay; P)) \longrightarrow И x.\left(\overline{ax} \parallel \left(ax; P\{^x/_y\}\right)\right)$$

The *select1* rule that instantiates the existential quantifier cannot permute with the *extrude name* rule. To permit such permutations, we may consider extending BV1 with the following rule.

$$C\{\ P\ \} \longrightarrow C\{\ \Im x.P\ \} \qquad \text{only if } x \# P \qquad \textit{(introduce name)}$$

The above rule would enable the above derivation to be transformed to the following derivation.

$$И x.\overline{ax} \parallel \exists y.(ay; P) \longrightarrow И x.\overline{ax} \parallel \Im x.\exists y.(ay; P) \longrightarrow И x.\overline{ax} \parallel \Im x.\left(ax; P\{^x/_y\}\right) \longrightarrow И x.\left(\overline{ax} \parallel \left(ax; P\{^x/_y\}\right)\right)$$

Observe that, in the above derivation, the *introduce name* rule considered, is used to eagerly declare that y is to be replaced by a private name. The *close* rule is then applied to link the private name with the private input. A similar, situation holds for permuting *select1* with the *extrude1*, which would require that the following rule is added to BV1.

$$C\{\ P\ \} \longrightarrow C\{\ \exists x.P\ \} \qquad \text{only if } x \# P \qquad \textit{(introduce1)}$$

This has the knock on effect of requiring the rules *close1* and *suspend1* from Fig. 6 for permuting down the *introduce1* rule. This leads us to the first attempt at a decomposition property for the *select* rule.

Proposition 11. *If $P \longrightarrow R$ in the core fragment of BV1, then there exists Q such that:*

- *$P \longrightarrow Q$ using only the rule select1 and additional rules introduce name and introduce1.*
- *$Q \longrightarrow R$ using the core rules without the rule select1 and extrude1, but with the additional rules close1 and suspend1.*

The above observations suggest an alternative rewrite system for BV1 presented for discussion in Fig. 6. A problem is that the *introduce1* and *introduce name* rules are not analytic in the sense that they introduce new operators, meaning that proof search is unwieldy. However, we know that the two axiomatisation of BV1 have equivalent power in terms of provable propositions. The proof is a consequence of cut elimination for BV1, Theorem 3.

Proposition 12. *BV1 and the alternative rewrite system in Fig. 6 are equivalent in expressive power.*

This brings us tantalisingly close to the decomposition results for BV extended with the exponentials NEL [25, 11]: where rules that introduce subterms are applied first (creation), then rules that preserve the size of terms (merging), followed by rules that remove subterms (destruction). We hypothesise that a similar decomposition is possible for the symmetric version of the system in Fig. 6, where for every rule of the form $C\{\ P\ \} \longrightarrow C\{\ Q\ \}$ we include a *co-rule* of the form $C\{\ \overline{Q}\ \} \longrightarrow C\{\ \overline{P}\ \}$, where the order

$$C\{\ \forall x.P \parallel \exists x.R\ \} \longrightarrow C\{\ \forall x.(P \parallel R)\ \}\ \text{(close1)} \qquad C\{\ \forall x.\mathrm{I}\ \} \longrightarrow C\{\ \mathrm{I}\ \}\ \text{(tidy1)}$$

$$C\{\ \exists x.P \odot \exists x.P\ \} \longrightarrow C\{\ \exists x.(P \odot Q)\ \}\ \text{(suspend1)} \qquad C\{\ \exists x.P\ \} \longrightarrow C\{\ P\{{}^{v}/_{x}\}\ \}\ \text{(select1)}$$

$$C\{\ \forall x.(P\ ;S)\ \} \longrightarrow C\{\ \forall x.P\ ;\forall x.S\ \}\ \text{(medial1)} \qquad C\{\ P\ \} \longrightarrow C\{\ \exists x.P\ \}\ \text{only if } x\ \#\ P\ \text{(introduce1)}$$

$$C\{\ Ии.P \parallel Э x.Q\ \} \longrightarrow C\{\ Ии.(P \parallel Q)\ \}\ \text{(close)} \qquad C\{\ Ии.\mathrm{I}\ \} \longrightarrow C\{\ \mathrm{I}\ \}\ \text{(tidy name)}$$

$$C\{\ Э x.P \odot Э x.S\ \} \longrightarrow C\{\ Э x.(P \odot S)\ \}\ \text{(suspend name)}$$

$$C\{\ Э x.P\ \} \longrightarrow C\{\ Ии.P\ \}\ \text{(fresh)} \qquad C\{\ Ии.(P\ ;S)\ \} \longrightarrow C\{\ Ии.P\ ;Ии.S\ \}\ \text{(medial new)}$$

$$C\{\ Ии.Э y.P\ \} \longrightarrow C\{\ Э y.Ии.P\ \}\ \text{(new wen)} \qquad C\{\ \forall x.Ъ y.P\ \} \longrightarrow C\{\ Ъ y.\forall x.P\ \}\ \text{(all name)}$$

$$C\{\ P\ \} \longrightarrow C\{\ Э x.P\ \}\ \text{only if } x\ \#\ P\ \text{(introduce name)}$$

Figure 6: Alternative rules for quantifiers for the purpose of discussing decomposition, where $\odot \in \{\parallel, ;\}$ and $Ъ \in \{Ии, Э, \exists\}$.

of the rewrite is reversed and the connectives are replaced with their dual connectives. For example the *co-sequence* rule discussed earlier is the co-rule for *sequence*.

Such a decomposition is however tricky since certain permutations can lead to a blow up in the size of proofs, that require ingenuity to control. The reader may wish to ponder the how the *co-sequence* rule and *introduce name* permute with each other, by using co-rules, where $x\ \#\ ((P \otimes Q)\ ;(R \otimes S)) \otimes W$.

$$C\{\ ((P \otimes Q)\ ;(R \otimes S)) \otimes W\ \} \longrightarrow C\{\ (P\ ;R) \otimes (Q\ ;S) \otimes W\ \} \longrightarrow C\{\ (P\ ;R) \otimes Э x.((Q\ ;S) \otimes W)\ \}$$

Fortunately, for the current work a more restricted decomposition property for existential quantifiers is sufficient. The decomposition applies only to derivations where the variables appearing in a term introduced by *select1* are never extruded.

Proposition 13. *Consider a term t and derivation in the core fragment of* BV1 *of the form* $\mathcal{L}\{\ \underline{\exists} x.P\ \} \longrightarrow \mathcal{L}'\{\ \underline{\exists} x.Q\ \}$*, where one existential quantifier is underlined. Also, assume that, for all variables y in the term t, there is never a rule in the derivation of the form* $C\{\ Ъ y.R \parallel \exists x.S\ \} \longrightarrow C\{\ Ъ y.(R \parallel \underline{\exists} x.S)\ \}$*. In such a scenario there is always a derivation of the following form in the core fragment of* BV1*:* $\mathcal{L}\{\ P\{{}^{t}/_{x}\}\ \} \longrightarrow \mathcal{L}'\{\ Q\{{}^{t}/_{x}\}\ \}$.

Observe that, if $\exists x.P$ is in a left context, it will never be the case that $\exists x.P$ is nested inside another existential quantifier; where a nested quantifier would be able to influence the the term introduced.

The above observations allows us to present a solution to *Scenario 1*. The derivation in the core fragment must be of the form.

$$T \parallel \left[\!\!\left[\mathcal{P}^0\Big\{\ \mathcal{P}^1\{\ \overline{x}z.P\ \} \parallel \mathcal{P}^2\{\ x(y).Q\ \}\ \Big\} \right]\!\!\right]_{\pi} \longrightarrow \mathcal{L}'\{\ \exists x.R\ \} \longrightarrow \mathcal{L}'\{\ R\{{}^{z}/_{x}\}\ \} \longrightarrow \mathcal{L}\{\ \overline{xz} \parallel xz\ \}$$

This can be transformed to a derivation in the core fragment of BV1 of the following form, where each $\mathcal{L}^i\{\ \}$ is structurally congruent to an embedding of the respective $\mathcal{P}^i\{\ \}$.

$$\begin{aligned} T \parallel \mathcal{L}^0\Big\{\ \mathcal{L}^1\{\ \overline{xz}\ ;[\![P]\!]_{\pi}\ \} \parallel \mathcal{L}^2\{\ \exists y.(xy\ ;[\![Q]\!]_{\pi})\ \}\ \Big\} \longrightarrow\ & T \parallel \mathcal{L}^0\Big\{\ \mathcal{L}^1\{\ \overline{xz}\ ;[\![P]\!]_{\pi}\ \} \parallel \mathcal{L}^2\Big\{\ xz\ ;\left[\!\!\left[Q\{{}^{z}/_{y}\} \right]\!\!\right]_{\pi}\ \Big\}\ \Big\} \\ \longrightarrow\ & \mathcal{L}'\{\ R\{{}^{z}/_{x}\}\ \} \longrightarrow \mathcal{L}\{\ \overline{xz} \parallel xz\ \} \end{aligned}$$

We can now delete the instances of the atoms $\overline{xz}$ and xz involved in the bottommost interaction everywhere in the proof, while removing redundant rules. This leads to a lazy proof of the following

proposition as required.

$$T \parallel \left[\!\left[\mathcal{P}^0\{\, \mathcal{P}^1\{\, P \,\} \parallel \mathcal{P}^2\{\, Q\{^{z}/_{y}\} \,\} \,\} \right]\!\right]_{\pi}$$

Scenario 3. Consider when both atoms appear in the embedding of a process such that the output name is locally bound. In this case, (up-to α-conversion of names) there exists a derivation in the core fragment of BV1, where x and z are not bound by $\mathcal{P}^1\{\ \}$, $\mathcal{P}^2\{\ \}$, or $\mathcal{P}^3\{\ \}$ and z is fresh for all processes appearing in the contexts $\mathcal{P}^1\{\ \}$ and $\mathcal{P}^3\{\ \}$.

$$T \parallel \left[\!\left[\mathcal{P}^0\{\, \mathcal{P}^1\{\, \nu z.\mathcal{P}^2\{\, \overline{x}z.P \,\} \,\} \parallel \mathcal{P}^3\{\, x(z).Q \,\} \,\} \right]\!\right]_{\pi} \longrightarrow \mathcal{L}\{\, \overline{xz} \parallel xz \,\}$$

Furthermore, there exists a lazy proof of $\mathcal{L}\{\, \overline{xz} \parallel xz \,\}$ in BV1.

Observe that for propositions of the above form there is always a τ transition.

Proposition 14. *For any π-calculus process of the form $\mathcal{P}^0\{\, \mathcal{P}^1\{\, \nu z.\mathcal{P}^2\{\, \overline{x}z.P \,\} \,\} \parallel \mathcal{P}^3\{\, x(z).Q \,\} \,\}$, where x and z are not bound by $\mathcal{P}^1\{\ \}$, $\mathcal{P}^2\{\ \}$, or $\mathcal{P}^3\{\ \}$ and z is fresh for all processes appearing in the contexts $\mathcal{P}^1\{\ \}$ and $\mathcal{P}^3\{\ \}$, there exists a τ transition to the process $\mathcal{P}^0\{\, \nu z.\big(\mathcal{P}^1\{\, \mathcal{P}^2\{\, P \,\} \,\} \parallel \mathcal{P}^3\{\, Q \,\}\big) \,\}$.*

The challenge with Scenario 3 is the binder for the private name being output should be extended such that the respective input action is in scope of the binder. This involves transforming the conclusion of the proof and the proof itself. Once such a transformation has been applied the above scenario is reduced to the same steps as in Scenario 2.

We firstly explain how the transformation works on an example. Suppose that we have the function over predicates P, where $\underline{\text{И}x}$ and $\underline{\text{Э}x}$ are distinguished binders, such that in a rule instance quantifiers underscored in the premise must be underscored in the conclusion. The underscore forces the paths of quantifiers through a proof to be identified. For example, the following derivation (part of a lazy proof) traces the path of a *new* connective through the structure, which is duplicated by the *medial name* rule associated with a *wen* quantifier though the *close* rule.

$$\begin{aligned}
\text{И}y.\underline{\text{И}x}.\big(\overline{ax}\,;\,\overline{by}\big) \parallel \underline{\text{Э}x}.ax \parallel \text{Э}y.by &\longrightarrow \text{И}y.\big(\underline{\text{И}x}.\overline{a}x\,;\,\underline{\text{И}x}.\overline{b}y\big) \parallel \underline{\text{Э}x}.ax \parallel \text{Э}y.by \\
&\longrightarrow \big(\text{И}y.\underline{\text{И}x}.\overline{a}x\,;\,\text{И}y.\underline{\text{И}x}.\overline{b}y\big) \parallel \underline{\text{Э}x}.ax \parallel \text{Э}y.by \\
&\longrightarrow \big(\big(\text{И}y.\underline{\text{И}x}.\overline{a}x \parallel \underline{\text{Э}x}.ax\big)\,;\,\text{И}y.\underline{\text{И}x}.\overline{b}y\big) \parallel \text{Э}y.by \\
&\longrightarrow \big(\text{И}y.\big(\underline{\text{И}x}.\overline{a}x \parallel \underline{\text{Э}x}.ax\big)\,;\,\text{И}y.\underline{\text{И}x}.\overline{b}y\big) \parallel \text{Э}y.by \\
&\longrightarrow \big(\text{И}y.\underline{\text{И}x}.(\overline{a}x \parallel ax)\,;\,\text{И}y.\underline{\text{И}x}.\overline{b}y\big) \parallel \text{Э}y.by \\
&\longrightarrow \big(\text{И}y.\underline{\text{И}x}.\text{I}\,;\,\text{И}y.\underline{\text{И}x}.\overline{b}y\big) \parallel \text{Э}y.by
\end{aligned}$$

If, to avoid variable capture, we assume that the variable x bound by the underscored quantifiers does not appear bound by another quantifier or appear free in the conclusion. The following function can be defined that removes underscored quantifiers from predicates, where $\kappa \in \{\alpha, \overline{\alpha}, \text{I}\}$, $\text{O} \in \{\forall, \text{И}, \text{Э}, \exists\}$ and $\odot \in \{\otimes, ;, \parallel\}$.

$$rm(\underline{\text{O}x}.P) = rm(P) \qquad rm(\text{O}y.P) = \text{O}y.rm(P) \qquad rm(P \odot Q) = rm(P) \odot rm(Q) \qquad rm(\kappa) = \kappa$$

The above function can be applied to the example derivation to obtain the derivation below. Observe the underscored quantifiers and redundant rules have been deleted and $\text{И}x$ is reinserted with a wider scope.

$$\begin{aligned}
\text{И}x.\big(\text{И}y.\big(\overline{ax}\,;\,\overline{by}\big) \parallel ax\big) \parallel \text{Э}y.by &\longrightarrow \text{И}x.\big(\big(\text{И}y.\overline{a}x\,;\,\text{И}y.\overline{b}y\big) \parallel ax\big) \parallel \text{Э}y.by \\
&\longrightarrow \text{И}x.\big((\text{И}y.\overline{a}x \parallel ax)\,;\,\text{И}y.\overline{b}y\big) \parallel \text{Э}y.by \\
&\longrightarrow \text{И}x.\big(\text{И}y.(\overline{a}x \parallel ax)\,;\,\text{И}y.\overline{b}y\big) \parallel \text{Э}y.by \\
&\longrightarrow \text{И}x.\big(\text{И}y.\text{I}\,;\,\text{И}y.\overline{b}y\big) \parallel \text{Э}y.by
\end{aligned}$$

The pattern illustrated above is generalised by the following proposition.

Proposition 15. *If* $\vdash C\{ P \}$*, where* P *may contain underscored quantifiers* $\underline{И x}$ *and* $\underline{\Im x}$ *connected by open and medial rules, such that* x *is only bound by underscored quantifiers and* $x \# C\{ P \}$*, then* $\vdash C\{ И x.rm(P) \}$*.*

A consequence of the above proposition for Scenario 3 is, for proofs of embedding of processes, we can expand the scope of quantifiers as required for scope extrusion. For increased generality, we aim to capture directly the labelled transition semantics of the π-calculus rather than the reduction semantics (which closes transitions up-to some structural congruence). For this reason, the above proposition does not automatically push scope to the maximum extent. For example, the π-calculus process $\nu y.(\nu x.\overline{a}x \parallel a(x))$ has a τ transition to $\nu y.\nu x.(1 \parallel 1)$ but strictly speaking has no τ transition to the trace equivalent process $\nu x.\nu y.(1 \parallel 1)$. This is reflected in the following example proof.

$$[\![\nu y.(\nu x.\overline{a}x \parallel a(x))]\!]_\pi = И y.(И x.\overline{ax} \parallel \exists x.ax) \longrightarrow И y.И x.(\overline{ax} \parallel \exists x.ax) \longrightarrow И y.И x.(\overline{ax} \parallel ax) \longrightarrow И y.И x.\mathrm{I}$$

Combined with the observations from Scenario 2, we have a procedure for constructing a proof of the following predicate, as required.

$$T \parallel \Big[\!\Big[\mathcal{P}^0\Big\{ \nu z.\Big(\mathcal{P}^1\Big\{ \mathcal{P}^2\{ P \} \Big\} \parallel \mathcal{P}^3\{ Q \}\Big) \Big\}\Big]\!\Big]_\pi$$

Scenario 4. Consider the scenario where the bottommost atomic interaction rule applies to an atom that can be traced to the output of a fresh name in the trace. In the core fragment there is a derivation of the following form, where T is a completed trace, Q is a π-calculus process, $\mathcal{L}\{\ \}$ is a left context, $\mathcal{P}^0\{\ \}$ and $\mathcal{P}^1\{\ \}$ are a process contexts that do not bind x and z; and z is fresh for $\mathcal{P}^0\{\ \}$.

$$\Im z.(xz\,;T) \parallel \Big[\!\Big[\mathcal{P}^0\Big\{ \nu z.\mathcal{P}^1\{ \overline{x}z.Q \} \Big\}\Big]\!\Big]_\pi \longrightarrow \mathcal{L}\{ xz \parallel \overline{xz} \}$$

Furthermore $\mathcal{L}\{ \mathrm{I} \}$ has a lazy proof. By applying the same observations and technique developed for Proposition 15, we can always pull the quantifier νz to the outermost level, removing corresponding *wen* operators to obtain a lazy proof of proposition $И z.\Big((xz\,;T) \parallel \Big[\!\Big[\mathcal{P}^0\Big\{ \mathcal{P}^1\{ \overline{x}z.Q \} \Big\}\Big]\!\Big]_\pi\Big)$. Furthermore, when $И z.P$ is provable P is provable, hence the outermost $И z$ can be deleted from the proof. Hence, by deleting the interacting atoms, we can always construct a proof of the proposition.

$$\vdash T \parallel \Big[\!\Big[\mathcal{P}^0\Big\{ \mathcal{P}^1\{ Q \} \Big\}\Big]\!\Big]_\pi$$

The process in the above proofs are always related by a labelled transition as follows.

Proposition 16. *Assume* Q *is a* π*-calculus process,* $\mathcal{L}\{\ \}$ *is a left context,* $\mathcal{P}^0\{\ \}$ *and* $\mathcal{P}^1\{\ \}$ *are a process contexts that do not bind* x *and* z*; and* z *is fresh for* $\mathcal{P}^0\{\ \}$*. There is always a* $\overline{x}[z]$ *transition from* $\mathcal{P}^0\Big\{ \nu z.\mathcal{P}^1\{ \overline{x}z.Q \} \Big\}$ *to* $\mathcal{P}^0\Big\{ \mathcal{P}^1\{ Q \} \Big\}$*.*

Scenario 5. The case for an input transition is similar to the case for the interaction of an input and free output. In this case, there is a derivation in the core fragment of the following form, where y and x are not bound by $\mathcal{P}\{\ \}$ and x does not appear free in $\mathcal{P}\{\ \}$.

$$(\overline{yz}\,;T) \parallel [\![\mathcal{P}\{ y(x).Q \}]\!]_\pi \longrightarrow \mathcal{L}\{ \overline{yz} \parallel yz \}$$

Assume also that $\mathcal{L}\{ \mathrm{I} \}$ has a lazy proof.

Observe that the above derivation in the core fragment must by of the following form.

$$(\overline{yz}\,;T) \parallel [\![\mathcal{P}\{ y(x).Q \}]\!]_\pi \longrightarrow \mathcal{L}'\{ \exists x.R \} \longrightarrow \mathcal{L}'\{ R\{^z/_x\} \} \longrightarrow \mathcal{L}\{ \overline{yz} \parallel yz \}$$

Hence by Proposition 13 there exists a derivation in the core fragment of the following form, where the first step is an instance of the *select1* rule, and $\mathcal{T}\{\ \}$ is the embedding of $\mathcal{P}\{\ \}$.

$$(\overline{yz}\,;T) \parallel [\![\mathcal{P}\{\ y(x).Q\ \}]\!]_\pi \longrightarrow (\overline{yz}\,;T) \parallel \mathcal{T}\{\ yz\,;[\![Q]\!]_\pi\ \} \longrightarrow \mathcal{L}'\{\ R\{^z/_x\}\ \}$$

By removing the pair of atoms $\overline{yz}$ and yz from the proof we construct a proof of $T \parallel [\![\mathcal{P}\{\ Q\{^z/_x\}\ \}]\!]_\pi$. As expected, this is always the right hand side of a labelled transition after applying substitution $\{^z/_x\}$, which represents the late instantiation of input variable x with value z. The following proposition verifies that indeed this is always the case.

Proposition 17. *Assuming x and y are not bound by $\mathcal{P}\{\ \}$ and x is fresh for $\mathcal{P}\{\ \}$, there is always a transition labelled with $y(x)$ from π-calculus process $\mathcal{P}\{\ y(x).Q\ \}$ to process $\mathcal{P}\{\ Q\ \}$.*

Scenario 6. There is also a scenario of no operational consequence where the bottommost rule in the interaction fragment is an instance of the *tidy name* rule in a left context. In this case, there exists a reduction $T \parallel [\![\mathcal{P}\{\ \nu x.1\ \}]\!]_\pi \longrightarrow \mathcal{L}\{\ Ux.\mathrm{I}\ \}$ in the core fragment of BV1 and a lazy proof of $\mathcal{L}\{\ \mathrm{I}\ \}$. In this case it is always possible to renormalise a lazy proof such that such an instance of the *tidy name* rule is delayed until the final rule of the proof. The case to take care of is where the *sequence* rule is applied as follows $\mathcal{L}\{\ Ux.\mathrm{I} \parallel P \parallel Q\ \} \longrightarrow \mathcal{L}\{\ (Ux.\mathrm{I} \parallel P)\,;Q\ \}$. In this scenario, renormalise the proof as follows $\mathcal{L}\{\ Ux.\mathrm{I} \parallel P \parallel Q\ \} \longrightarrow \mathcal{L}\{\ P\,;(Ux.\mathrm{I} \parallel Q)\ \}$ or remove the *sequence* rule if $P \equiv \mathrm{I}$.

4.3 Extracting Executions from Renormalised Proofs

The following proposition summarises the observations made above.

Proposition 18 (progress)**.** *For traces T, and π-calculus process R, if $\vdash T \parallel [\![R]\!]_\pi$ then at least one of the following holds:*

- *$T = \mathrm{I}$ and $R\checkmark$.*
- *$R = \mathcal{P}^0\{\ \mathcal{P}^1\{\ \overline{y}z.P\ \} \parallel \mathcal{P}^2\{\ y(x).Q\ \}\ \}$, where y and z are not bound by $\mathcal{P}^1\{\ \}$ or $\mathcal{P}^2\{\ \}$, and the following holds: $\vdash T \parallel [\![\mathcal{P}^0\{\ \mathcal{P}^1\{\ P\ \} \parallel \mathcal{P}^2\{\ Q\{^z/_x\}\ \}\ \}]\!]_\pi$.*
- *$R = \mathcal{P}^0\{\ \mathcal{P}^1\{\ Uz.\mathcal{P}^2\{\ \overline{x}z.P\ \}\ \} \parallel \mathcal{P}^3\{\ x(z).Q\ \}\ \}$, where x and z are not bound by $\mathcal{P}^1\{\ \}$, $\mathcal{P}^2\{\ \}$, or $\mathcal{P}^3\{\ \}$, z is fresh for all processes appearing in the contexts $\mathcal{P}^1\{\ \}$ and $\mathcal{P}^3\{\ \}$, and the following holds: $\vdash T \parallel [\![\mathcal{P}^0\{\ Uz.(\mathcal{P}^1\{\ \mathcal{P}^2\{\ P\ \}\ \} \parallel \mathcal{P}^3\{\ Q\ \})\ \}]\!]_\pi$.*
- *$T = \overline{xy}\,;U$ and $R = \mathcal{P}\{\ x(z).P\ \}$ such that x and z are not bound by $\mathcal{P}\{\ \}$, and the following holds: $\vdash U \parallel [\![\mathcal{P}\{\ P\{^y/_x\}\ \}]\!]_\pi$.*
- *$T = xz\,;U$ and $R = \mathcal{P}\{\ \overline{x}z.P\ \}$ such that x and z are not bound by $\mathcal{P}\{\ \}$, and the following holds: $\vdash U \parallel [\![\mathcal{P}\{\ P\ \}]\!]_\pi$.*
- *$T = \exists z.\,(xz\,;U)$ and $R = \mathcal{P}^0\{\ \nu z.\mathcal{P}^1\{\ \overline{x}z.P\ \}\ \}$, where x and z are not bound by $\mathcal{P}^0\{\ \}$ or $\mathcal{P}^1\{\ \}$, z is fresh in $\mathcal{P}^0\{\ \}$, and the following holds: $\vdash U \parallel [\![\mathcal{P}^0\{\ \mathcal{P}^1\{\ P\ \}\ \}]\!]_\pi$.*

The above proposition is of independent interest as a progress property. We can consider the completed trace to be a restricted session type for the process in the following sense. When the process is executed such that the session type is respected, we are guaranteed that the process can always make progress until it successfully terminates without deadlock.

The main contribution of this paper, Proposition 4, follows from the above proposition.

Proof of Proposition 4. Assume that T is a trace and P is a process such that $\vdash T \parallel [\![P]\!]_\pi$. Proceed by induction on the structure of the trace. In the base, case $T = \mathbb{1}$ hence by Proposition 18 it must be the case that $P\checkmark$ hence, by definition, P has completed trace $\mathbb{1}$.

Assume that $T = \Im z.(xz\,;U)$. By Proposition 18 there are two possibilities, either $P \xrightarrow{\tau} Q$ such that $\vdash T \parallel [\![Q]\!]_\pi$ or $P \xrightarrow{\overline{x}[z]} R$ and $\vdash U \parallel [\![R]\!]_\pi$. Since τ transitions strictly decrease the size of the process P only finitely many τ transitions can be applied; after which, for some P' and R the output transition $P' \xrightarrow{\overline{x}[y]} R$ must be enabled and $\vdash U \parallel [\![R]\!]_\pi$. By applying the induction hypothesis, R has completed trace U; so, P' has completed trace $\Im z.(xz\,;U)$. Therefore, since P' was preceded by τ transitions only P has completed trace T. A similar argument holds for input and free output transitions.

Recall the above Proposition 4 was the missing link in proving the soundness of linear implication with respect to completed trace inclusion. Thereby linear implication can be used confidently as a pre-order over processes, without being concerned that two processes become related that were not related by completed trace inclusion.

5 The Versatility of the Processes-as-Predicates Approach

In this section we highlight the versatility of our approach in two ways. Firstly, we highlight that processes of other calculi, not only the π-calculus, can be embedded as processes in *BV1* such that implication defines a sound process preorder. Secondly, we discuss a surprising strict inequality in BV1 and how it arises naturally when input and private input coexist in a process calculus.

5.1 Extending Results to the Internal π-calculus

We briefly outline how the techniques in this paper can be extended to the internal π-calculus [23], called the πI-calculus, where inputs are guaranteed to be private. The private restriction on input names is enforced by using $\Im$ in place of $\exists$ in the embedding of private inputs. We are able to obtain that, for our embedding of πI-calculus processes as predicates in BV1, linear implication is sound with respect to completed trace inclusion.

The embedding of πI-calculus processes as BV1 propositions is defined as follows.

$$[\![1]\!]_{\pi I} = \mathbb{1} \qquad [\![P \parallel Q]\!]_{\pi I} = [\![P]\!]_{\pi I} \parallel [\![Q]\!]_{\pi I} \qquad [\![\nu x.P]\!]_{\pi I} = \text{И}x.[\![P]\!]_{\pi I}$$

$$[\![x[z]\,.P]\!]_{\pi I} = \Im z.(xz\,;[\![P]\!]_{\pi I}) \qquad [\![\overline{x}[z]\,.P]\!]_{\pi I} = \text{И}z.(\overline{xz}\,;[\![P]\!]_{\pi I})$$

Recall from the discussion in Section 3.2 that we use the syntax $x[z]$ to represent *private input* in the πI-calculus to syntactically disambiguate from the semantically distinct input $x(z)$ in the π-calculus. The output process $\overline{x}[z].P$ behaves much like the output $\nu z.\overline{x}z.P$ in the π-calculus, where the private name binder νz appears immediately before the action that outputs the name z.

Well-formed completed traces can be defined independently of processes, as follows.

Definition 11. *The unit $\mathbb{1}$ is a completed trace and, inductively if T is a completed trace then:*

- $\text{И}y.(\overline{xy}\,;T)$ *is a completed trace, where $x \neq y$.*
- $\Im y.(xy\,;T)$ *is a completed trace, where $x \neq y$.*

Successfully terminated processes in πI-calculus are defined such that $1\checkmark$ as for the π-calculus. Completed traces are associated with processes such that if $P\checkmark$ then P has completed trace $\mathbb{1}$ and, inductively, if Q has completed trace T then the following hold:

- *If $P \xrightarrow{x[z]} Q$ then P has completed trace $\text{И}z.(\overline{xz}\,;T)$.*

$$
\begin{array}{lll}
P ::= 1 & & \text{(success)} \\
\quad | \ \nu x.P & & \text{(nu)} \\
\quad | \ \overline{x}[z].P & & \text{(private input)} \\
\quad | \ x[z].P & & \text{(private output)} \\
\quad | \ P \parallel P & & \text{(par)} \\
\\
A ::= \tau \mid \overline{x}[z] \mid x[z] & & \text{(actions)}
\end{array}
$$

$$
\frac{}{\overline{x}[z].P \xrightarrow{\overline{x}[z]} P} \qquad \frac{}{x[z].P \xrightarrow{\overline{x}[z]} P} \qquad \frac{P \xrightarrow{A} Q}{\nu x.P \xrightarrow{A} \nu x.Q}\ x \notin \mathrm{n}(A)
$$

$$
\frac{P \xrightarrow{A} Q}{P \parallel R \xrightarrow{A} Q \parallel R}\ \begin{array}{l}\text{if } A = \overline{x}[z] \\ \text{or } A = x[z] \\ \text{then } z \# R\end{array} \qquad \frac{P \xrightarrow{\overline{x}[z]} P' \quad Q \xrightarrow{x[z]} Q'}{P \parallel Q \xrightarrow{\tau} \nu z.(P' \parallel Q')}
$$

Figure 7: Syntax and labelled transition system for πI-calculus, plus the symmetric rules for parallel composition. Function $n(.)$ is such that $n(x[z]) = n(\overline{x}[z]) = \{x, z\}$ and $n(\tau) = \emptyset$. Assertion $x \# P$ is such that x is fresh for P where $z[x].P$, $\overline{z}[x].P$ and $\nu x.P$ bind x in P.

- *If* $P \xrightarrow{\overline{x}[z]} Q$ *then* P *has completed trace* $\Im z.(xz\,;T)$.
- *If* $P \xrightarrow{\tau} Q$ *then* P *has completed trace* T.

We can establish a progress result for πI-calculus processes and their completed traces. The proof uses Propositions 8 and 15, by a similar argument to Scenarios 3 and 4 for the π-calculus.

Proposition 19 (progress)**.** *For πI traces T, and πI-calculus process R, if $\vdash T \parallel [\![R]\!]_{\pi I}$ then at least one of the following holds:*

- $T = \mathbb{1}$ *and* $R\checkmark$.
- $R = \mathcal{P}^0\{\, \mathcal{P}^1\{\, \overline{x}[z].P \,\} \parallel \mathcal{P}^2\{\, x[z].Q \,\} \,\}$, *where x and z are not bound by $\mathcal{P}^1\{\ \}$ or $\mathcal{P}^2\{\ \}$, and the following holds:* $\vdash T \parallel \left[\!\!\left[\mathcal{P}^0\{\, \nu z.\left(\mathcal{P}^1\{\, P \,\} \parallel \mathcal{P}^2\{\, Q \,\}\right) \,\}\right]\!\!\right]_{\pi I}$.
- $T = \Im z.\,(xz\,;U)$ *and* $R = \mathcal{P}\{\, \overline{x}[z].P \,\}$, *where x and z are not bound by $\mathcal{P}\{\ \}$, and z is fresh in $\mathcal{P}\{\ \}$, and* $\vdash U \parallel [\![\mathcal{P}\{\, P \,\}]\!]_{\pi I}$.
- $T = \mathit{Wz}.\,(\overline{xz}\,;U)$ *and* $R = \mathcal{P}\{\, x[z].P \,\}$, *where x and z are not bound by $\mathcal{P}\{\ \}$, and z is fresh in $\mathcal{P}\{\ \}$, and* $\vdash U \parallel [\![\mathcal{P}\{\, P \,\}]\!]_{\pi I}$.

By the above progress result, we can establish the following main result in this paper for the πI-calculus, similar to Proposition 4 for π-calculus. The proposition enables us to extract successful executions for proofs.

Proposition 20. *Assuming T is a πI-calculus completed trace and P is a process in the πI-calculus, if $\vdash T \parallel [\![P]\!]_{\pi I}$ then P has completed trace T.*

The converse direction for the π-calculus follows by induction over the structure of the labelled transition system and cut elimination for BV1, Theorem 3.

Proposition 21. *Assuming T is a πI-calculus completed trace and P is a process in the πI-calculus, if P has completed trace T then $\vdash T \parallel [\![P]\!]_{\pi I}$.*

From the above results and Theorem 3, we have thereby established the soundness of linear implication with respect to trace inclusion for the πI-calculus.

Theorem 5. *If $\vdash P \multimap Q$ then, for all completed traces T, if P has completed trace T, then Q has completed trace T.*

5.2 Explaining Curiosities Regarding Name Extrusion

A curiosity of BV1 is that names can only be extruded outwards in general. By this we mean that, when $x \# Q$, there is not a proof of $И x.P \parallel Q \multimap И x.(P \parallel Q)$, i.e. $И x.(P \parallel Q) \mathrel{-\!\!\circ\!\!\!+} И x.P \parallel Q$.

A logical reason for this restriction is that attempts to include an equivalence equating the predicates $И x.P \parallel Q$ and $И x.(P \parallel Q)$, where $x \# Q$, presented difficulties when proving cut elimination. Most likely, this is for deeper reasons. To see this, observe that, with the above rewrite rule, the implication $[\![\nu z.\overline{a}z.\nu z.\overline{a}z]\!]_\pi \multimap [\![\nu z.\overline{a}z.\overline{a}z]\!]_\pi$ would **wrongly** be provable. Such an implication would be **unsound** with respect to trace inclusion. The former process outputs two distinct names, but the later cannot.

This curiosity can also be explained due to the ability of BV1 to express both the π-calculus and the πI-calculus. To see this, suppose that we aim to provide a reduction semantics that combines the π-calculus and the πI-calculus. Such a semantics should give a meaning to process $a[x].a[y] \parallel \nu z.\overline{a}z.\overline{a}z$, where $a[x].a[y]$ here represents private inputs of the πI-calculus and $\nu z.\overline{a}z.\overline{a}z$ is π-calculus term that outputs the same fresh name twice. Naively, we may **wrongly** allow the following terminating reduction.

$$a[x].a[y] \parallel \nu z.\overline{a}z.\overline{a}z \xrightarrow{\tau} \nu z.(a[y] \parallel \overline{a}z) \not\equiv a[y] \parallel \nu z.\overline{a}z \xrightarrow{\tau} \nu z.1$$

The problem with the above is that we have broken the contract on the first private input. In particular, the fresh name z unified with x must be guaranteed freshness in the scope of x. Intuitively, the process conducting a private input promises never to inadvertently reveal the private name. Unfortunately, the reverse name extrusion indicated by $\not\equiv$ forgets this contract and inputs the same "fresh" name instead of a properly distinct name. To fix the above, a reduction semantics for a combined $\pi/\pi I$-calculus should only (lazily) push fresh names outwards, as with the *extrude name* rule in BV1.

Fortunately, the labelled transitions system naturally only pushes fresh names outward lazily. To see this, observe that when $a[y]$ is private input of the πI-calculus this process can only perform a transition labelled with private input $a[y]$ and not the more general input $a(y)$. Therefore, $\nu z.(a[y] \parallel \overline{a}z)$ is deadlocked since $\overline{a}z$ cannot guarantee the output is private in the relevant scope. In contrast, the following labelled transition to a successfully terminated state is permitted.

$$\dfrac{\dfrac{}{a[y] \xrightarrow{a[z]} 1} \qquad \dfrac{\dfrac{}{\overline{a}z \xrightarrow{\overline{a}z} 1}}{\nu z.\overline{a}z \xrightarrow{\overline{a}[z]} 1}}{a[y] \parallel \nu z.\overline{a}z \xrightarrow{\tau} \nu z.(1 \parallel 1)}$$

Thus processes $\nu z.(a[y] \parallel \overline{a}z)$ and $a[y] \parallel \nu z.\overline{a}z$ are not completed trace equivalent (nor bisimilar), in a π-calculus extended with an explicitly private input. Although, we do know that any completed trace of $\nu z.(a[y] \parallel \overline{a}z)$, such as $Э z.(az\,; И y.\overline{ay})$, is a completed trace of $a[y] \parallel \nu z.\overline{a}z$ by extending the results of this paper to such a combined $\pi/\pi I$-calculus.

6 Conclusion

The main objective achieved was to complete the proof of the soundness of linear implication in BV1 with respect to trace inclusion for the π-calculus and πI-calculus. The missing link compared to previous work [12] was Proposition 4. The proof of this results is discussed throughout Section 4.

The added value of this paper is that more general observation were made, suggesting several lines of equiry. In terms of proof theory we provide an example to show that for any extension of the whole of BV, such as BV1, is impossible to limit ourselves to "lazy" proofs, where interaction rules apply only in certain context. However, we observe that the lazy proofs cover fragments sufficient for embedding a range of process calculi. It appears that, as in linear logic [2, 3], there are fragments where the proof

search space is more controlled than for the whole proof system — see for instance the discussion around Propositions 8, 13 and 15. If every rule is restricted to a left context, not only interaction, the search space is narrowed further. Such control of search space is perpendicular to other techniques developed for taming proof search in the calculus of structures [14].

Another observation is that that Propositions 18 and 19, that are key to Propositions 4 and 20, are essentially progress properties. This reinforces the perspective that the calculus of structures provides a foundation for session types, presented in related work [8]. In the session type literature, we expect that if a process is well typed it is always possible to make a transition to a state that is also well typed, but possibly with respect to an updated type. Since we know that more general predicates than traces can be assigned lazy proofs, by Proposition 8, future work could extend such a progress result with more general session types generalising completed traces.

We also observed that a sound and complete semantics, based on homomorphisms between series-parallel pomsets [7], provides a tighter sound and complete match for linear implication. It is an open problem to develop an extension of the pomset semantics BV (without *times*) to a pomset semantics for a fragment of BV1 (also without *times*) such that the subtleties of names are fully respected.

Towards a philosophical explanation of *proof* in terms of *success*. We have established a precise correspondence relating proofs of completed traces and processes to successful executions of processes. In the world of linear logic, which BV1 conservatively extends, there is no notion of truth, only provability. Equating provability and success, is a digestible alternative to equating provability and truth in the classical world. Indeed Blass [5] had an intuition based on sucessful executions of protocols when developing game semantics for linear logic. Note that success can be more general than successfully terminating. E.g., the following predicate we expect to be provable and to correspond to two interacting recursive processes that can be successfully executed forever without terminating (under certain progress assumptions):

$$\nu X.\,(a\,;X) \parallel \nu Y.\,(\overline{a}\,;Y)$$

In the above, ν is a greatest fixed point operator, which can be used to encode recursion in processes.

Future work includes proving cut elimination for second order extensions of MAV1 to model recursive processes, and *delegation* as found in session types. The basic techniques for proving cut elimination for the additives [12] extends to the first order quantifiers [13]. However, these techniques do not directly extend to the second order case. In particular, in the first-order case, a context lemma is employed where the induction measure does not account for the size of terms in a process. In contrast, in the second order case instantiations of existential quantifiers with predicates can result in a non trivial blow up in the size of the context, in which second order variables may occur. Either techniques are required to control the blow up of proofs or an entirely different strategy to cut elimination is required.

References

[1] Samson Abramsky. Proofs as processes. *Theor. Comput. Sci.*, 135(1):5–9, 1994.

[2] Jean-Marc Andreoli. Logic programming with focusing proofs in linear logic. *Journal of Logic and Computation*, 2(3):297–347, 1992.

[3] Jean-Marc Andreoli and Remo Pareschi. Linear objects in a logic processes with built-in inheritance. In *Logic Programming, Proceedings of the Seventh International Conference, Jerusalem, Israel, June 18-20, 1990*, pages 495–510, 1990.

[4] Jesper Bengtson and Joachim Parrow. Formalising the pi-calculus using nominal logic. *Logical Methods in Computer Science*, 5(2), 2009.

[5] Andreas Blass. A game semantics for linear logic. *Annals of Pure and Applied logic*, 56(1):183–220, 1992.

[6] Paola Bruscoli. A purely logical account of sequentiality in proof search. In *ICLP*, volume 2401 of *LNCS*, pages 302–316. Springer, 2002.

[7] Gabriel Ciobanu and Ross Horne. A provenance tracking model for data updates. In *Proceedings 11th International Workshop on Foundations of Coordination Languages and Self Adaptation, FOCLASA 2012, Newcastle, U.K., September 8, 2012.*, pages 31–44, 2012.

[8] Gabriel Ciobanu and Ross Horne. Behavioural analysis of sessions using the calculus of structures. In *PSI 2015, 25-27 August, Kazan, Russia*, volume 9609 of *LNCS*, pages 91–106, 2015.

[9] Jay L. Gischer. The equational theory of pomsets. *Theor. Comput. Sci.*, 61:199–224, 1988.

[10] Alessio Guglielmi. A system of interaction and structure. *ACM Transactions on Computational Logic*, 8(1), 2007.

[11] Alessio Guglielmi and Lutz Straßburger. A system of interaction and structure V: The exponentials and splitting. *Math. Struct. Comp. Sci.*, 21(03):563–584, 2011.

[12] Ross Horne. The consistency and complexity of multiplicative additive system virtual. *Sci. Ann. Comp. Sci.*, 25(2):245–316, 2015.

[13] Ross Horne, Alwen Tiu, Bogdan Aman, and Gabriel Ciobanu. Private names in non-commutative logic. In *CONCUR 2016*, pages 31:1–31:16. LIPIcs, 2016.

[14] Ozan Kahramanogullari. Interaction and depth against nondeterminism in proof search. *Logical Methods in Computer Science*, 10(2), 2014.

[15] Ozan Kahramanogullari. True concurrency of deep inference proofs. In *Logic, Language, Information, and Computation - 23rd International Workshop, WoLLIC 2016, Puebla, Mexico, August 16-19th, 2016. Proceedings*, pages 249–264, 2016.

[16] Patrick Lincoln and Andre Scedrov. First-order linear logic without modalities is nexptime-hard. *Theor. Comput. Sci.*, 135(1):139–153, 1994.

[17] Raymond McDowell, Dale Miller, and Catuscia Palamidessi. Encoding transition systems in sequent calculus. *Theor. Comput. Sci.*, 294(3):411–437, 2003.

[18] Dale Miller. The pi-calculus as a theory in linear logic: Preliminary results. In *Extensions of Logic Programming, Third International Workshop, ELP'92, Bologna, Italy, February 26-28, 1992, Proceedings*, volume 660 of *Lecture Notes in Computer Science*, pages 242–264. Springer, 1993.

[19] Dale Miller and Alwen Tiu. A proof theory for generic judgements. *ACM Transactions on Computational Logic (TOCL)*, 6(4):749–783, 2005.

[20] Robin Milner, Joachim Parrow, and David Walker. A calculus of mobile processes, I and II. *Information and Computation*, 100(1):1–77, 1992.

[21] Vaughan Pratt. Modelling concurrency with partial orders. *International Journal of Parallel Programming*, 15(1):33–71, 1986.

[22] Christian Retoré. Pomset logic: A non-commutative extension of classical linear logic. In *TLCA'97*, volume 1210 of *LNCS*, pages 300–318. Springer, 1997.

[23] Davide Sangiorgi. π-calculus, internal mobility, and agent-passing calculi. *Theoretical Computer Science*, 167(1):235–274, 1996.

[24] Alex K. Simpson. Sequent calculi for process verification: Hennessy-milner logic for an arbitrary GSOS. *J. Log. Algebr. Program.*, 60-61:287–322, 2004.

[25] Lutz Straßburger and Alessio Guglielmi. A system of interaction and structure IV: the exponentials and decomposition. *TOCL*, 12(4):23, 2011.

[26] Alwen Tiu. A system of interaction and structure II: The need for deep inference. *Logical Methods in Computer Science*, 2(2:4):1–24, 2006.

[27] Alwen Tiu and Dale Miller. Proof search specifications of bisimulation and modal logics for the π-calculus. *TOCL*, 11(2):13, 2010.

[28] Daniele Varacca and Nobuko Yoshida. Typed event structures and the linear pi-calculus. *Theor. Comput. Sci.*, 411(19):1949–1973, 2010.