Private Names in Non-Commutative Logic

Ross Horne[1], Alwen Tiu[1], Bogdan Aman[2], and Gabriel Ciobanu[2]

1 **School of Computer Science and Engineering, Nanyang Technological University, Singapore**
rhorne@ntu.edu.sg, atiu@ntu.edu.sg
2 **Romanian Academy, Institute of Computer Science, Blvd. Carol I no.8, 700505 Iaşi, Romania**
bogdan.aman@iit.academiaromana-is.ro, gabriel@info.uaic.ro

Abstract

We present an expressive but decidable first-order system (named MAV1) defined by using the calculus of structures, a generalisation of the sequent calculus. In addition to first-order universal and existential quantifiers the system incorporates a de Morgan dual pair of nominal quantifiers called 'new' and 'wen', distinct from the self-dual Gabbay-Pitts and Miller-Tiu nominal quantifiers. The novelty of the operators 'new' and 'wen' is they are polarised in the sense that 'new' distributes over positive operators while 'wen' distributes over negative operators. This greater control of bookkeeping enables private names to be modelled in processes embedded as predicates in MAV1. Modelling processes as predicates in MAV1 has the advantage that linear implication defines a precongruence over processes that fully respects causality and branching. The transitivity of this precongruence is established by novel techniques for handling first-order quantifiers in the cut elimination proof.

1998 ACM Subject Classification F.4.1 Mathematical Logic; F.3.2 Semantics of Programming Languages; F.1.2 Modes of Computation

Keywords and phrases process calculi, calculus of structures, nominal logic, causal consistency

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2016.31

1 Introduction

This paper introduces a first-order non-commutative logic, of significance for modelling processes, expressed in a formalism called the calculus of structures [17, 18, 34, 36, 37]. The calculus of structures is effectively a term rewriting system modulo a congruence that can express proof systems combining connectives for sequentiality and parallelism. The calculus of structures was motivated by a desire to understand why pomset logic [30] could not be expressed in the sequent calculus. Pomset logic is inspired by pomsets [29] and linear logic [15], the former being a model of concurrency respecting causality [33], while the latter can be interpreted in various ways as a logic of resources and concurrency [9, 21, 42]. Acknowledging connections between pomsets, linear logic and concurrency, it is natural to consider the calculus of structures in the context of concurrency theory.

Initial investigations [7] relate a basic process calculus with parallel composition ($P \parallel Q$) and action prefix ($\alpha.P$) to a proof system in the calculus of structures called BV [17]. In that work, a logical characterisation of completed traces is established using provability. As a consequence of this logical characterisation and cut elimination for BV, if P implies Q and P has a completed trace then Q has the same completed trace. Thereby, linear implication is strictly finer than completed trace inclusion. Strictness follows since some processes related by linear implication are not related by trace inclusion. For example, a desirable property of linear implication is that *autoconcurrency* [4, 40, 41] is avoided, since the embedding of $\alpha \parallel \alpha$ does **not** logically imply the embedding of $\alpha.\alpha$. Avoiding autoconcurrency indicates that linear implication fully respects the causal order of events. Preorders that respect causality are significant for various applications, not limited to soundly reasoning about

processes deployed on large high-availability distributed systems where a consensus on interleavings is infeasible but causality is respected [11, 22].

Recently, BV was extended with additives to obtain the system MAV [20], enabling choice in processes to be modelled. Results concerning multi-party compatibility and subtyping in a session type system inspired by Scribble [19] have been established using MAV [12]. The current paper continues this line of enquiry by modelling name passing processes in a conservative extension of MAV with first-order quantifiers, named MAV1. The system MAV1 is also a conservative extension of first-order multiplicative additive linear logic MALL1 [23] with mix. A novelty is that MAV1 includes a pair of de Morgan dual nominal quantifiers pronounced *new* and *wen* and written И and Э respectively. In a processes-as-predicates embedding [16], И models new name restriction in the π-calculus. The dual nominal quantifier Э is essential for defining linear implication and models the input of private names in a processes-as-predicates embedding for the πI-calculus [32].

Logically speaking, nominal quantifiers И and Э sit between $\forall$ and $\exists$ such that $\forall x.P \multimap И x.P$ and $И x.P \multimap Э x.P$ and $Э x.P \multimap \exists x.P$, where $\multimap$ is linear implication. The quantifier И is similar in some respects to $\forall$, whereas Э is similar to $\exists$. A crucial difference between $\exists x.P$ and $Э x.P$ is that variable x in the latter cannot be instantiated with arbitrary terms, but only 'fresh' names introduced by И.

The need for new quantifiers. We illustrate why neither universal quantification nor an established self-dual nominal quantifier [14, 25, 28] are capable of soundly modelling name restriction in a processes-as-predicates embedding. We argue that, since trace inclusion is considered to be amongst the coarsest preorders on process [39], it makes sense to impose a minimum requirement that linear implication cannot relate two processes that are unrelated by trace inclusion.

In the following, observe that $R_1 = \nu x.(\overline{a}x \parallel \overline{b}x)$ is a π-calculus process that can output a fresh name twice, once on channel a and once on channel b; but cannot output two perceivably distinct names in any execution. In contrast, observe that $R_2 = \nu x.\overline{a}x \parallel \nu x.\overline{b}x$ is a π-calculus process that outputs two distinct fresh names before terminating, but cannot output the same name twice in any execution. The processes R_1 and R_2 are unrelated by trace inclusion in either direction.

For an encoding using universal quantifiers for name restriction, processes R_1 and R_2 are respectively encoded as predicates $P_1 = \forall x.\big(\overline{ax} \parallel \overline{bx}\big)$ and $P_2 = \forall x.\overline{ax} \parallel \forall x.\overline{bx}$, where operator $\parallel$ is overloaded to connect parallel composition and *par* from linear logic. Unfortunately, the implication $P_2 \multimap P_1$ is provable. However, R_2 can output two perceivably distinct names but R_1 cannot, so implication would not be sound with respect to trace inclusion. Additionally, we must also avoid the following *diagonalisation* property [25]: $\forall x.\forall y.P(x, y) \multimap \forall z.P(z, z)$.

The self-dual nominal quantifiers of either Gabbay-Pitts [28] or Miller-Tiu [25, 14], as recently investigated in the calculus of structures [31], do successfully avoid the above diagonalisation property. Unfortunately, rather surprisingly, encoding private names using any of these self-dual nominal quantifiers, say ∇, leads to the following problem. Suppose processes R_1 and R_2 are encoded by the respective predicates $Q_1 = \nabla x.(\overline{ax} \parallel \overline{bx})$ and $Q_2 = \nabla x.\overline{ax} \parallel \nabla x.\overline{bx}$. In this case, the linear implication $Q_1 \multimap Q_2$ is provable. This implication is also unsound, since R_1 has a trace that outputs two identical names, whereas R_2 admits no such trace.

Our *new* quantifier И, distinct from the Gabbay-Pitts operator, addresses the above limitations of universal quantification and established self-dual nominal quantifiers. In addition to avoiding diagonalisation, our И quantifier does not distribute over parallel composition in either direction. In MAV1, the predicates $И x.(\overline{ax} \parallel \overline{bx})$ and $И x.\overline{ax} \parallel И x.\overline{bx}$ are, correctly, unrelated by linear implication.

Outline. For a new logical system it is necessary to justify correctness, which we approach in proof theoretic style by cut elimination. Section 2 defines MAV1 and explains cut elimination. Section 3 illustrates a processes-as-predicates embedding in MAV1 and explains that linear implication defines a branching-time preorder that respects causality. Section 4 presents a more detailed explanation of the rules for the nominal quantifiers and the novel strategy of the cut elimination proof.

2 Syntax and Semantics of Predicates in MAV1

In this section we present the syntax and semantics of a first-order system expressed in the calculus of structures, with the technical name MAV1. We assume that the reader has a basic understanding of term-rewriting systems [24].

A term-rewriting system requires an *abstract syntax*, defined in Fig. 1. The *rewrite rules*, in Fig 3, define rules that can be applied to rewrite a predicate of the form on the left of the long right arrow to the predicate on the right. All rewrite rules can be applied in any context, i.e. MAV1 predicates from Fig. 1 with a hole of the following form $C\{\ \} ::= \{\ \cdot\ \} \mid C\{\ \} \odot P \mid P \odot C\{\ \} \mid \mathcal{O}x.C\{\ \}$, where $\odot \in \{;, \parallel, \otimes, \&, \oplus\}$ and $\mathcal{O} \in \{\exists, \forall, И, Э\}$.

Further to rewriting according to rules, the term-rewriting system is defined *modulo a congruence*, where a congruence is an equivalence relation that holds in any context. The congruence, defined in Fig. 2, makes *par* and *times* commutative and *seq* non-commutative in general. The congruence enables α-conversion for quantifiers. In addition, *equivariance* allows names bound by successive nominal quantifiers to be swapped.

As standard, we define a freshness predicate such that a variable 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. We also assume the standard notion of capture avoiding substitution of a variable for a term. Terms may be constructed from variables, constants and function symbols. When predicates model process in the π-calculus, atoms are pairs of terms, where the first term represents a channel and the second a message.

We postpone a discussion on the rules until after we introduce the notion of a proof and explain cut elimination in the next section.

P ::=	α	(atom)
	$\overline{\alpha}$	(co-atom)
	1	(unit)
	$\forall x.P$	(all)
	$\exists x.P$	(some)
	$Иx.P$	(new)
	$Эx.P$	(wen)
	$P \,\&\, P$	(with)
	$P \oplus P$	(plus)
	$P \parallel P$	(par)
	$P \otimes P$	(times)
	$P\ ;\ P$	(seq)

Figure 1 *MAV1* syntax.

2.1 Linear Implication and Cut Elimination

This section confirms that MAV1 is a consistent logical system, as established by a cut elimination theorem. Surprisingly, to date, the only direct proof of cut elimination involving quantifiers in the calculus of structures is for a self-dual nominal quantifier [31] distinct from any quantifier in MAV1. Related cut elimination results involving first-order quantifiers in the calculus of structures rely on a correspondence with the sequent calculus [6, 35]. However, due to the presence of the non-commutative operator *seq* there is no sequent calculus presentation [37] for MAV1; hence we pursue here a direct proof.

A derivation is a sequence of zero or more rewrite rules from Fig. 3, where the congruence in Fig. 2 can be applied at any point. We are particularly interested in special derivations, called proofs.

▶ **Definition 1.** A *proof* in MAV1 is a derivation $P \longrightarrow 1$ from a predicate P to the unit 1. When such a derivation exists, we say that P is provable, and write $\vdash P$.

$(P, \parallel, 1)$ and $(P, \otimes, 1)$ are commutative monoids $\quad \mathcal{O}x.P \equiv \mathcal{O}y.(P\{^{y}/_{x}\})$ if $y \# \mathcal{O}x.P$ (α-conversion)

$(P, ;, 1)$ is a monoid $\quad Иx.Иy.P \equiv Иy.Иx.P$ (equivariance) $\quad Эx.Эy.P \equiv Эy.Эx.P$ (equivariance)

Figure 2 Congruence ($\equiv$) for MAV1 predicates. For α-conversion $\mathcal{O} \in \{\exists, \forall, И, Э\}$ is any quantifier.

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

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

$$C\{\ (P \,\&\, Q) \parallel R\ \} \longrightarrow C\{\ (P \parallel R) \,\&\, (Q \parallel R)\ \} \quad \text{(external)} \qquad C\{\ \mathrm{I} \,\&\, \mathrm{I}\ \} \longrightarrow C\{\ \mathrm{I}\ \} \quad \text{(tidy)}$$

$$C\{\ (P\ ;\ Q) \,\&\, (R\ ;\ S)\ \} \longrightarrow C\{\ (P \,\&\, R)\ ;\ (Q \,\&\, S)\ \} \quad \text{(medial)}$$

$$C\{\ P \oplus Q\ \} \longrightarrow C\{\ P\ \} \quad \text{(left choice)} \qquad C\{\ P \oplus Q\ \} \longrightarrow C\{\ Q\ \} \quad \text{(right choice)}$$

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

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

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

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

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

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

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

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

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

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

$$C\{\ Ↄ x.P \,\&\, Ↄ x.S\ \} \longrightarrow C\{\ Ↄ x.(P \,\&\, S)\ \} \text{ for } Ↄ \in \{И, Э\} \quad \text{(with name)}$$

$$C\{\ Ↄ x.P \,\&\, R\ \} \longrightarrow C\{\ Ↄ x.(P \,\&\, R)\ \} \text{ only if } x \# R \text{ for } Ↄ \in \{И, Э\} \quad \text{(left name)}$$

$$C\{\ R \,\&\, Ↄ x.Q\ \} \longrightarrow C\{\ Ↄ x.(R \,\&\, Q)\ \} \text{ only if } x \# R \text{ for } Ↄ \in \{И, Э\} \quad \text{(right name)}$$

Figure 3 Rewrite rules for predicates in system MAV1. Notice the figure is divided into four parts. The first part defines sub-system BV [17]. The first and second parts together define sub-system MAV [20]. The following restrictions are placed on the rules to ensure the system is analytic [8].

- The *switch*, *sequence*, *medial1*, *medial new* and *extrude new* rules are such that $P \not\equiv \mathrm{I}$ and $S \not\equiv \mathrm{I}$.
- The *medial* rule is such that either $P \not\equiv \mathrm{I}$ or $R \not\equiv \mathrm{I}$ and also either $Q \not\equiv \mathrm{I}$ or $S \not\equiv \mathrm{I}$.
- The rules *external*, *extrude1*, *extrude new*, *left wen* and *right wen* are such that $R \not\equiv \mathrm{I}$.

A *derivation* is a sequence of rewrites, where the congruence in Fig. 2 can be applied at any point in a derivation. The length of a derivation involving only the congruence is zero. The length of a derivation involving one rule from Fig. 3 is one. Given a derivation $P \longrightarrow Q$ of length m and another $Q \longrightarrow R$ of length n, the derivation $P \longrightarrow R$ is of length $m + n$. Unless we make it clear in the context that we refer to a specific rule, $\longrightarrow$ is generally used to represent derivations of any length.

To explore the theory of proofs, two auxiliary definitions are introduced: linear negation and linear implication. Notice in the syntax in Fig. 1 linear negation applies only to atoms.

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

$$\overline{\overline{\alpha}} = \alpha \qquad \overline{P \otimes Q} = \overline{P} \parr \overline{Q} \qquad \overline{P \parr Q} = \overline{P} \otimes \overline{Q} \qquad \overline{P \oplus Q} = \overline{P} \,\&\, \overline{Q} \qquad \overline{P \,\&\, Q} = \overline{P} \oplus \overline{Q}$$
$$\overline{1} = \bot \qquad \overline{P \,;\, Q} = \overline{P} \,;\, \overline{Q} \qquad \overline{\forall x.P} = \exists x.\overline{P} \qquad \overline{\exists x.P} = \forall x.\overline{P} \qquad \overline{\nu x.P} = \mathfrak{B} x.\overline{P} \qquad \overline{\mathfrak{B} x.P} = \nu x.\overline{P}$$

Linear implication, written $P \multimap Q$, is defined as $\overline{P} \parr Q$.

Linear negation defines de Morgan dualities. As in linear logic, the multiplicatives $\otimes$ and $\parr$ are de Morgan dual; as are the additives $\&$ and $\oplus$, the first-order quantifiers $\exists$ and $\forall$, and the nominal quantifiers ν and $\mathfrak{B}$. As in BV, sequential composition and the unit are self-dual.

The following proposition is simply a reflexivity property of linear implication in MAV1.

▶ **Proposition 3** (Reflexivity)**.** *For any predicate P, $\vdash \overline{P} \parr P$ holds.*

The main result of this paper is the following, which is a generalisation of *cut elimination* to the setting of the calculus of structures.

▶ **Theorem 4** (Cut elimination)**.** *For any predicate P, if $\vdash C\{ P \otimes \overline{P} \}$ holds, then $\vdash C\{\bot\}$ holds.*

The above theorem can be stated alternatively by supposing that there is a proof in MAV1 extended with the extra rewrite rule: $C\{\bot\} \longrightarrow C\{ P \otimes \overline{P} \}$ (cut). Given such a proof, a new proof can be constructed that uses only the rules of MAV1. In this formulation, we say that *cut* is *admissible*.

The cut elimination proof for the propositional sub-system MAV appears in a companion journal paper [20]. The current paper advances cut-elimination techniques to tackle first-order system MAV1, as achieved by the lemmas in later sections. Before proceeding with the necessary lemmas, we provide a corollary that demonstrates that one of many consequences of cut elimination is indeed that linear implication defines a precongruence — a reflexive transitive relation that holds in any context.

▶ **Corollary 5.** *Linear implication defines a precongruence.*

3 Linear Implication as a Precongruence for Processes-as-Predicates

We highlight connections between MAV1 and the π-calculus. This illustrates the rationale behind design decisions in MAV1. We assume the reader is familiar with the syntax of the π-calculus. For the π-calculus define a processes-as-predicates embedding as follows.

$$[\![x(y).P]\!]_\pi = \exists y.(xy \,;\, [\![P]\!]_\pi) \qquad [\![\overline{x}y.P]\!]_\pi = \overline{xy} \,;\, [\![P]\!]_\pi \qquad [\![P \parallel Q]\!]_\pi = [\![P]\!]_\pi \parr [\![Q]\!]_\pi$$
$$[\![\nu x.P]\!]_\pi = \mathfrak{B} x.[\![P]\!]_\pi \qquad [\![P + Q]\!]_\pi = [\![P]\!]_\pi \oplus [\![Q]\!]_\pi \qquad [\![1]\!]_\pi = \bot$$

Notice action prefixes are captured using atoms consisting of pairs of first-order variables. We consider preorders that do not observe τ actions and are *termination sensitive* [1, 41], hence distinguish between successful termination and deadlock. Successful termination is indicated by a process 1, differing from 0 typical of process calculi. For example $P + 1$ represents the process that may behave like P or may successfully terminate, contrasting to $P + 0$ which only may only proceed as P. This distinction is useful for modelling protocols; for example, we can choose to perform no action in certain executions to avoid deadlocking. Furthermore, 1 as a primitive process matches the unit inherited from BV. Otherwise, the semantics are standard for the π-calculus.

Define labelled transitions for the π-calculus by the following deductive system, plus the symmetric rules for parallel composition and choice. Function $n(.)$ is such that $n(x(y)) = n(\overline{x}(y)) = n(\overline{x}y) = \{x, y\}$, and $x \# P$ is such that x is fresh for P where $z(x).P$ and $\nu x.P$ bind x in P.

$$\frac{}{x(y).P \xrightarrow{x(y)} P} \qquad \frac{}{\overline{x}y.P \xrightarrow{\overline{x}y} P} \qquad \frac{P \xrightarrow{A} Q}{P + R \xrightarrow{A} Q} \qquad \frac{}{P + 1 \xrightarrow{\tau} 1}$$

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

$$\frac{P \xrightarrow{\overline{x}y} Q}{\nu y.P \xrightarrow{\overline{x}(y)} Q}\ x \neq y \qquad \frac{P \xrightarrow{\overline{x}(z)} P' \quad Q \xrightarrow{x(z)} Q'}{P \parallel Q \xrightarrow{\tau} \nu z.(P' \parallel Q')}\ \dagger \qquad \frac{P \xrightarrow{\overline{x}y} P' \quad Q \xrightarrow{x(z)} Q'}{P \parallel Q \xrightarrow{\tau} P' \parallel Q'\{^{y}/_{z}\}}$$

Define (symbolic weak) completed traces inductively as follows.

- For any process P formed using only the unit 1, name restriction and parallel composition (i.e. with no actions or choice) P has the completed trace I.
- If $P \xrightarrow{x(y)} Q$ and Q has completed trace tr then P has completed trace $\forall y.(\overline{xy}\,;\, tr)$.
- If $P \xrightarrow{\overline{x}(y)} Q$ and Q has completed trace tr then P has completed trace $\Im y.(xy\,;\, tr)$.
- If $P \xrightarrow{\overline{x}y} Q$ and Q has completed trace tr then P has completed trace $xy\,;\, tr$.
- If $P \xrightarrow{\tau} Q$ and Q has completed trace tr then P has completed trace tr.

Observe that deadlocked processes have no completed trace. Contrast for example $\nu x.x(y) \parallel 1$ and $\nu x.x(y) + 1$, where the former has no completed trace but the later has completed trace I.

Interestingly, *equivariance*, in Fig. 2, is a design decision in the sense that cut elimination is still provable for a MAV1 without *equivariance*. However, *equivariance* is a requirement for modelling private names in process calculi. Consider π-calculus process $\nu y.\nu x.(\overline{z}x.\overline{w}y)$ and the completed trace $\Im x.(zx\,;\, \Im y.wy)$ that outputs a fresh name on channel z then a separate fresh name on channel w. $\vdash \mathit{И}y.\mathit{И}x.(\overline{zx}\,;\, \overline{wy}) \parallel \Im x.(zx\,;\, \Im y.wy)$ is provable only with *equivariance*. Hence *equivariance* is necessary for the following proposition.

▶ **Proposition 6.** *If a process P has completed trace tr then $\vdash [\![P]\!]_\pi \parallel tr$.*

Proof. The proof follows by induction over the structure of the derivation for a labelled transition. We present only two inductive cases, for the communication of an input and bound output (†), and the extrusion of a bound output over a distinct private name (‡).

Assume the induction hypotheses, $\vdash [\![P]\!]_\pi \parallel \forall z\left(\overline{xy}\,;\, \overline{[\![P']\!]_\pi}\right)$ and $\vdash [\![Q]\!]_\pi \parallel \Im z\left(xz\,;\, \overline{[\![Q']\!]_\pi}\right)$. Implication $\vdash \left([\![P]\!]_\pi \parallel \forall z.\left(\overline{xz}\,;\, \overline{[\![P']\!]_\pi}\right)\right) \otimes \left([\![Q]\!]_\pi \parallel \Im z.\left(xz\,;\, \overline{[\![Q']\!]_\pi}\right)\right) \multimap [\![P]\!]_\pi \parallel [\![Q]\!]_\pi \parallel \Im z.\left(\overline{[\![P']\!]_\pi} \otimes \overline{[\![Q']\!]_\pi}\right)$ is provable as follows, using Proposition 3.

$$\begin{array}{ll}
\left(\overline{[\![P]\!]_\pi} \otimes \exists z.(xz\,;\, [\![P']\!]_\pi)\right) \parallel \left(\overline{[\![Q]\!]_\pi} \otimes \mathit{И}z.(\overline{xz}\,;\, [\![Q']\!]_\pi)\right) \parallel [\![P]\!]_\pi \parallel [\![Q]\!]_\pi \parallel \Im z.\left(\overline{[\![P']\!]_\pi} \otimes \overline{[\![Q']\!]_\pi}\right) & \\
\longrightarrow \left(\left(\overline{[\![P]\!]_\pi} \parallel [\![P]\!]_\pi\right) \otimes \exists z\,(xz\,;\, [\![P']\!]_\pi)\right) \parallel \left(\left(\overline{[\![Q]\!]_\pi} \parallel [\![Q]\!]_\pi\right) \otimes \mathit{И}z\,(\overline{xz}\,;\, [\![Q']\!]_\pi)\right) \parallel \Im z.\left(\overline{[\![P']\!]_\pi} \otimes \overline{[\![Q']\!]_\pi}\right) & \\
\longrightarrow \exists z.(xz\,;\, [\![P']\!]_\pi) \parallel \mathit{И}z.(\overline{xz}\,;\, [\![Q']\!]_\pi) \parallel \Im z.\left(\overline{[\![P']\!]_\pi} \otimes \overline{[\![Q']\!]_\pi}\right) & \text{switch and Proposition 3} \\
\longrightarrow \exists z.(xz\,;\, [\![P']\!]_\pi) \parallel \mathit{И}z.\left((\overline{xz}\,;\, [\![Q']\!]_\pi) \parallel \left(\overline{[\![P']\!]_\pi} \otimes \overline{[\![Q']\!]_\pi}\right)\right) & \text{close} \\
\longrightarrow \mathit{И}z.\left(\exists z.(xz\,;\, [\![P']\!]_\pi) \parallel (\overline{xz}\,;\, [\![Q']\!]_\pi) \parallel \left(\overline{[\![P']\!]_\pi} \otimes \overline{[\![Q']\!]_\pi}\right)\right) & \text{extrude new} \\
\longrightarrow \mathit{И}z.\left((xz\,;\, [\![P']\!]_\pi) \parallel (\overline{xz}\,;\, [\![Q']\!]_\pi) \parallel \left(\overline{[\![P']\!]_\pi} \otimes \overline{[\![Q']\!]_\pi}\right)\right) & \text{select1} \\
\longrightarrow \mathit{И}z.\left((xz \parallel \overline{xz})\,;\, \left([\![P']\!]_\pi \parallel [\![Q']\!]_\pi \parallel \left(\overline{[\![P']\!]_\pi} \otimes \overline{[\![Q']\!]_\pi}\right)\right)\right) & \text{sequence} \\
\longrightarrow \mathit{И}z.\left((xz \parallel \overline{xz})\,;\, \left(\left([\![P']\!]_\pi \parallel \overline{[\![P']\!]_\pi}\right) \otimes \left([\![Q']\!]_\pi \parallel \overline{[\![Q']\!]_\pi}\right)\right)\right) & \text{switch} \\
\longrightarrow \mathit{И}z.\mathrm{I} \longrightarrow \mathrm{I} & \text{Proposition 3 and tidy new}
\end{array}$$

Hence by Theorem 4, $\vdash [\![P \parallel Q]\!]_\pi \parallel \overline{[\![\nu z.(P' \parallel Q')]\!]_\pi}$, as required.

As the induction hypothesis, assume that $\vdash [\![P]\!]_\pi \parallel \Im z.\left(xz\,;\overline{[\![Q]\!]_\pi}\right)$ holds, where y is such that $x \neq y$ and $z \neq y$. Thereby the following proof can be constructed directly as required.

$$\begin{array}{rcll}
[\![\nu y.P]\!]_\pi \parallel \Im z.\left(xz\,;\overline{[\![\nu y.Q]\!]_\pi}\right) = & И y.[\![P]\!]_\pi \parallel \Im z.\left(xz\,;\Im y.\overline{[\![Q]\!]_\pi}\right) & & \text{by definition} \\
\longrightarrow & И y.[\![P]\!]_\pi \parallel \Im z.\Im y.\left(xz\,;\overline{[\![Q]\!]_\pi}\right) & & \text{right wen} \\
\equiv & И y.[\![P]\!]_\pi \parallel \Im y.\Im z.\left(xz\,;\overline{[\![Q]\!]_\pi}\right) & & \text{equivariance} \\
\longrightarrow & И y.\left([\![P]\!]_\pi \parallel \Im z.\left(xz\,;\overline{[\![Q]\!]_\pi}\right)\right) \longrightarrow И y.\mathrm{I} \longrightarrow \mathrm{I} & & \text{induction and tidy}
\end{array}$$

The proof concludes by inductively applying Theorem 4 to each transition forming a trace. □

The above proposition also holds for a processes-as-predicates embedding for the πI-calculus [32], where input of private names is such that $[\![x(y).P]\!]_{\pi I} = \Im y.(xy\,;[\![P]\!]_{\pi I})$ and output of private names is such that $[\![\overline{x}(y).P]\!]_{\pi I} = И y.(\overline{xy}\,;[\![P]\!]_{\pi I})$. The labelled transition system for the πI-calculus, is such that $\overline{x}(y).P \xrightarrow{\overline{x}(y)} P$ and complete traces are such that, if $P \xrightarrow{x(y)} Q$ and Q has completed trace tr then P has completed trace $И y.(\overline{xy}\,;tr)$. We envision models of processes exploiting more of the expressive power of MAV1, such as the primitives employed for modelling session types using MAV [20].

For a basic process calculus with only parallel composition and prefix the converse to Proposition 6 is known to hold [7]. The converse direction for the π-calculus relies on techniques beyond cut elimination, hence will receive separate attention in a future paper.

Linear implication v.s. completed traces. We re-emphasise that the above completed trace semantics is a minimal justification for design decisions. Completed traces and linear implication are at the opposite ends of the linear-time/branching-time [39] and interleaving/causal [33] spectra.

A characteristic distinction between linear-time and branching-time preorders is in how processes of the form $\alpha.(P + Q)$ and $\alpha.P + \alpha.Q$ are related. For completed traces they are equivalent but for linear implication, only the direction $\vdash [\![\alpha.(P + Q)]\!]_\pi \multimap [\![\alpha.P + \alpha.Q]\!]_\pi$ holds. Hence linear implication is at the branching-time end of the spectrum.

Simulation preorders are also at the branching-time end of the spectrum. However, many simulation preorders interleave events as characterised by *expansion* rules [26] which equate processes with identical interleavings. For linear implication, expansion holds in one direction only. For example, processes $\alpha.\alpha$ and $\alpha \parallel \alpha$ have the same interleavings, but linear implication holds only in the direction $\vdash [\![\alpha.\alpha]\!]_\pi \multimap [\![\alpha \parallel \alpha]\!]_\pi$. As a further example, processes $\alpha.(\alpha.\beta \parallel \beta)$ and $\alpha.\beta \parallel \alpha.\beta$ have identical interleavings but linear implication holds only in the direction established by the following proof.

$$\begin{array}{rcll}
[\![\alpha.(\alpha.\beta \parallel \beta)]\!]_\pi \multimap [\![\alpha.\beta \parallel \alpha.\beta]\!]_\pi = & \left(\overline{\alpha}\,;\left(\left(\overline{\alpha}\,;\overline{\beta}\right)\otimes\overline{\beta}\right)\right) \parallel (\alpha\,;\beta) \parallel (\alpha\,;\beta) & & \text{by definition} \\
\longrightarrow & \left((\overline{\alpha} \parallel \alpha)\,;\left(\left(\left(\overline{\alpha}\,;\overline{\beta}\right)\otimes\overline{\beta}\right) \parallel \beta\right)\right) \parallel (\alpha\,;\beta) & & \text{sequence} \\
\longrightarrow & \left(\left(\overline{\alpha}\,;\overline{\beta}\right)\otimes\overline{\beta}\right) \parallel \beta \parallel (\alpha\,;\beta) & & \text{interaction} \\
\longrightarrow & \left(\overline{\alpha}\,;\overline{\beta}\right)\otimes\left(\overline{\beta} \parallel \beta\right) \parallel (\alpha\,;\beta) & & \text{switch} \\
\longrightarrow & \left(\overline{\alpha}\,;\overline{\beta}\right) \parallel (\alpha\,;\beta) & & \text{interaction} \\
\longrightarrow & (\overline{\alpha} \parallel \alpha)\,;\left(\overline{\beta} \parallel \beta\right) & & \text{sequence} \\
\longrightarrow & \mathrm{I} & & \text{interaction}
\end{array}$$

By not identifying interleavings, we argue that linear implication respects the causal order of events. Existing notions of simulation that respect the causal order of events, such as history preserving simulation [4, 40, 41], have not yet been extended to the setting with private names. Thereby MAV1 enables us to objectively explore the properties of fresh names in this part of the spectrum. For example, the implication $\vdash [\![\nu x.\,(P \parallel Q\{^y/_z\})]\!]_\pi \multimap [\![\nu x.(\overline{x}y.P \parallel x(z).Q)]\!]_\pi$ is provable. However, the converse is not provable. Intuitively, although no other process can communicate on channel x, the processes can be distinguished by a network partition interrupting communication the private channel.

Observe that no bisimulation can be complete with respect to logical equivalence. To see this observe that the embeddings $[\![\alpha.(\beta + \gamma) + \alpha.\beta]\!]_\pi$ and $[\![\alpha.(\beta + \gamma)]\!]_\pi$ are logically equivalent; whereas

any bisimulation distinguishes these processes. Logical equivalence, as with *mutual simulation* [39], is checked using a preorder in each direction, by proving predicate $(P \multimap Q)$ & $(Q \multimap P)$.

Questions formally relating observational preorders and linear implication rely on cut elimination for MAV1. This leads us to the cut elimination result of this paper.

4 Logical Properties of the Pair of Nominal Quantifiers

This section offers deeper insight into the nature of the nominal quantifiers И and Э, and sketches the cut elimination proof.[1] Cut elimination (Theorem 4) is achieved by advancing methods developed in the propositional sub-system MAV [20]. A direct proof of *co-rule elimination* for universal quantifiers (Lemma 11) simplifies *splitting* (Lemma 16), and *context reduction* (Lemma 20) is adapted for existential quantifiers by working up-to substitutions. Lemmas check the interplay between nominal quantifiers and other connectives, as illustrated by the discussion in the following subsection.

4.1 Discussion on the Rules for Nominal Quantifiers

The rules for the nominal quantifiers *new* И and *wen* Э require some justification. The *close* and *tidy name* rules ensure the reflexivity of implication for nominal quantifiers. Using the *extrude new* rule (and Proposition 3) we can establish the following proof, and its dual statement $\vdash \Im x.P \multimap \exists x.P$.

$$\forall x.P \multimap И x.P = \exists x.\overline{P} \parallel И x.P \longrightarrow И x.\Big(\exists x.\overline{P} \parallel P\Big) \longrightarrow И x.\Big(\overline{P} \parallel P\Big) \longrightarrow И x.\mathrm{I} \longrightarrow \mathrm{I}$$

Using the *fresh* rule we can establish the following implication between *new* and *wen*.

$$И x.P \multimap \Im x.P = \Im x.\overline{P} \parallel \Im x.P \longrightarrow И x.\overline{P} \parallel \Im x.P \longrightarrow И x.\Big(\overline{P} \parallel P\Big) \longrightarrow И x.\mathrm{I} \longrightarrow \mathrm{I}$$

This completes the chain $\vdash \forall x.P \multimap И x.P$, $\vdash И x.P \multimap \Im x.P$ and $\vdash \Im x.P \multimap \exists x.P$. These linear implications are strict unless $x \# P$, in which case, for $\mathcal{O} \in \{\forall, \exists, И, \Im\}$, $\mathcal{O}x.P$ is logically equivalent to P. For example, using the *fresh* and *extrude new* rules, the following holds given $x \# P$.

$$И x.P \multimap P = \Im x.\overline{P} \parallel P \longrightarrow И x.\overline{P} \parallel P \longrightarrow И x.\Big(\overline{P} \parallel P\Big) \longrightarrow И x.\mathrm{I} \longrightarrow \mathrm{I}$$

Alternatively, the *extrude new* and *fresh* rules could be replaced by extending the congruence with $И x.P \equiv P \equiv \Im x.P$, where $x \# P$. However, this has the disadvantage that an arbitrary number of nominal quantifiers can be introduced during proof search thereby jeopardising analyticity [8].

The *medial new* rule is particular to handling nominal quantifiers in the presence of the self-dual non-commutative operator *seq*. To see why this rule cannot be excluded, consider the following.

$$(a \,;\, \Im x.(b \,;\, c)) \otimes (d \,;\, \Im x.(e \,;\, f)) \multimap (a \,;\, \exists x.b \,;\, \exists x.c) \otimes (d \,;\, \exists x.e \,;\, \exists x.f)$$
$$(a \,;\, \exists x.b \,;\, \exists x.c) \otimes (d \,;\, \exists x.e \,;\, \exists x.f) \multimap ((a \,;\, \exists x.b) \otimes (d \,;\, \exists x.e)) \,;\, (\exists x.c \otimes \exists x.f)$$

Without using the *medial new* rule, the above predicates are provable but the following predicate would not be provable; hence cut elimination cannot hold without the *medial new* rule.

$$(a \,;\, \Im x.(b \,;\, c)) \otimes (d \,;\, \Im x.(e \,;\, f)) \multimap ((a \,;\, \exists x.b) \otimes (d \,;\, \exists x.e)) \,;\, (\exists x.c \otimes \exists x.f)$$

[1] Details of proofs appear in a technical report at: `http://iit.iit.tuiasi.ro/TR/reports/fml1502.pdf`

In contrast, with the *medial new* rule the above predicate is provable, verified by the following proof.

$$
\begin{aligned}
&\left(\overline{a}\,;\, И x.\left(\overline{b}\,;\,\overline{c}\right)\right) \parallel \left(\overline{d}\,;\, И x.\left(\overline{e}\,;\,\overline{f}\right)\right) \parallel ((a\,;\,\exists x.b)\otimes(d\,;\,\exists x.e))\,;\,(\exists x.c\otimes\exists x.f)\\
&\quad\longrightarrow\quad \left(\overline{a}\,;\, И x.\overline{b}\,;\, И x.\overline{c}\right) \parallel \left(\overline{d}\,;\, И x.\overline{e}\,;\, И x.\overline{f}\right) \parallel ((a\,;\,\exists x.b)\otimes(d\,;\,\exists x.e))\,;\,(\exists x.c\otimes\exists x.f)\\
&\quad\longrightarrow\quad \left(\left(\overline{a}\,;\, И x.\overline{b}\right) \parallel \left(\overline{d}\,;\, И x.\overline{e}\right)\right)\,;\,\left(И x.\overline{c} \parallel И x.\overline{f}\right) \parallel ((a\,;\,\exists x.b)\otimes(d\,;\,\exists x.e))\,;\,(\exists x.c\otimes\exists x.f)\\
&\quad\longrightarrow\quad \left(\left(\overline{a}\,;\, И x.\overline{b}\right) \parallel \left(\overline{d}\,;\, И x.\overline{e}\right) \parallel ((a\,;\,\exists x.b)\otimes(d\,;\,\exists x.e))\right)\,;\,\left(И x.\overline{c} \parallel И x.\overline{f} \parallel (\exists x.c\otimes\exists x.f)\right)\\
&\quad\longrightarrow\quad \left(\left(\left(\overline{a}\,;\, И x.\overline{b}\right) \parallel (a\,;\,\exists x.b)\right)\otimes\left(\left(\overline{d}\,;\, И x.\overline{e}\right) \parallel (d\,;\,\exists x.e)\right)\right)\,;\,\left((И x.\overline{c} \parallel \exists x.c)\otimes\left(И x.\overline{f} \parallel \exists x.f\right)\right)\\
&\quad\longrightarrow\quad \left(\left((\overline{a} \parallel a)\,;\,\left(И x.\overline{b} \parallel \exists x.b\right)\right)\otimes\left(\left(\overline{d} \parallel d\right)\,;\,(И x.\overline{e} \parallel \exists x.e)\right)\right)\,;\,\left((И x.\overline{c} \parallel \exists x.c)\otimes\left(И x.\overline{f} \parallel \exists x.f\right)\right)\\
&\quad\longrightarrow\quad \left(\left((\overline{a} \parallel a)\,;\, И x.\left(\overline{b} \parallel \exists x.b\right)\right)\otimes\left(\left(\overline{d} \parallel d\right)\,;\, И x.(\overline{e} \parallel \exists x.e)\right)\right)\,;\,\left(И x.(\overline{c} \parallel \exists x.c)\otimes И x.\left(\overline{f} \parallel \exists x.f\right)\right)\\
&\quad\longrightarrow\quad \left(\left((\overline{a} \parallel a)\,;\, И x.\left(\overline{b} \parallel b\right)\right)\otimes\left(\left(\overline{d} \parallel d\right)\,;\, И x.(\overline{e} \parallel e)\right)\right)\,;\,\left(И x.(\overline{c} \parallel c)\otimes И x.\left(\overline{f} \parallel f\right)\right)\\
&\quad\longrightarrow\quad (И x.\mathrm{I}\otimes И x.\mathrm{I})\,;\,(И x.\mathrm{I}\otimes И x.\mathrm{I})\longrightarrow \mathrm{I}
\end{aligned}
$$

Notice that the above proof uses only the *medial new*, *extrude new* and *tidy name* rules for nominals. These three rules are of the same form as the rules for universal quantifiers, hence the same argument holds for the necessity of the *medial1* rule.

Including the *medial new* rule forces the *medial wen* rule to be included. To see this observe that $(И x.a\,;\,И x.b)\otimes(И x.c\,;\,И x.d)\multimap И x.(a\,;\,b)\otimes И x.(c\,;\,d)$ and $И x.(a\,;\,b)\otimes И x.(c\,;\,d)\multimap И x.((a\,;\,b)\otimes(c\,;\,d))$ are provable. However, without the *medial wen* rule the following implication is not provable, which would contradict the main cut elimination result of this paper.

$$(И x.a\,;\,И x.b)\otimes(И x.c\,;\,И x.d)\multimap И x.((a\,;\,b)\otimes(c\,;\,d))$$

Fortunately, including the *medial wen* rule ensures that the above implication is provable. A similar argument justifies the inclusion of the *left wen* and *right wen* rules.

As with focussed proof search [3, 10], assigning a positive or negative polarity to operators explains certain distributivity properties. Consider $\parallel$, &, $\forall$ and И to be negative operators, and $\otimes$, $\oplus$, $\exists$ and Э to be positive operators, where *seq* is neuter. The negative quantifier И distributes over all positive operators. Considering positive operator *times* for example, $\vdash И x.\alpha\otimes И x.\beta\multimap И x.\,(\alpha\otimes\beta)$ holds but the converse implication does not hold. Furthermore, assuming x appears free in α and β, $Э x.\alpha\otimes Э x.\beta$ and $Э x.\,(\alpha\otimes\beta)$ are unrelated by linear implication. Dually, for the negative operator *par* the only distributivity property that holds for nominal quantifiers is $\vdash Э x.\,(\alpha\parallel\beta)\multimap Э x.\alpha\parallel Э x.\beta$. The *new wen* rule completes this picture of *new* distributing over positive operators and *wen* distributing over negative operators. From the perspective of embedding name-passing process calculi in logic, the above distributivity properties of *new* and *wen* suggest that processes should be encoded using negative operators И and $\parallel$ for private names and parallel composition (or perhaps dually, using positive operators Э and $\otimes$), so as to avoid private names distributing over parallel composition, which we have shown to be problematic in the introduction.

The control of distributivity exercised by *new* and *wen* contrast with the situation for universal and existential quantifiers, where $\exists$ commutes in one direction over all operators and $\forall$ commutes with all operators in the opposite direction. For a system with *equivariance* some of these distributivity properties for И over & and $\forall$ are explicit rules *all name*, *with name*, *left name*, *right name*. These rules allow И quantifiers to propagate to the front of certain contexts. In the sense of control of distributivity, *new* and *wen* behave more like multiplicatives than additives, but are unrelated to multiplicative quantifiers in the logic of bunched implications [27].

4.2 Preliminary Lemmas and Killing Contexts

We extend a trick employed for MAV [20] to MAV1 where *with* & and *all* $\forall$ receive a more direct treatment than other operators. The proof for *with* has a knock on effect for the nominal quantifiers

requiring some vacuous *new* and *wen* quantifiers to be removed; while the proof for *all* requires closure of rules under substitution of terms for variables. This leads to the following five lemmas, established directly by functions over predicates.

▶ **Lemma 7** (Substitution). *If $P \longrightarrow Q$ holds then $P\{^{v}/_{x}\} \longrightarrow Q\{^{v}/_{x}\}$ holds.*

▶ **Lemma 8** (Vacuous new). *If $\vdash C\{\ \text{И}x.P\ \}$ holds and $x \# P$ then $\vdash C\{\ P\ \}$ holds.*

▶ **Lemma 9** (Vacuous wen). *If $\vdash C\{\ \Im x.P\ \}$ holds and $x \# P$ then $\vdash C\{\ P\ \}$ holds.*

▶ **Lemma 10** (Branching). *If $\vdash C\{\ P \,\&\, Q\ \}$ holds then both $\vdash C\{\ P\ \}$ and $\vdash C\{\ Q\ \}$ hold.*

▶ **Lemma 11** (Universal). *If $\vdash C\{\ \forall x.P\ \}$ holds then, for all terms v, $\vdash C\{\ P\{^{v}/_{x}\}\ \}$ holds.*

We require a restricted form of context called a killing context (terminology is from [10]). A killing context is a context with one or more holes, defined as follows.

▶ **Definition 12.** A killing context is a context defined by the following grammar.

$$\mathcal{T}\{\ \} ::= \{\ \cdot\ \} \mid \mathcal{T}\{\ \} \,\&\, \mathcal{T}\{\ \} \mid \forall x.\mathcal{T}\{\ \} \mid \text{И}x.\mathcal{T}\{\ \}$$

In the above, $\{\ \cdot\ \}$ is a hole into which any predicate can be plugged. An n-ary killing context is a killing context in which n holes appear.

A killing context represents a context that cannot in general be removed until all other rules in a proof have been applied, hence the corresponding *tidy* rules are suspended until the end of a proof. A killing context has properties that are applied frequently in proofs, characterised by the following lemma.

▶ **Lemma 13.** *For any killing context $\mathcal{T}\{\ \}$, $\vdash \mathcal{T}\{\ \text{I}, \ldots, \text{I}\ \}$ holds; and, assuming the variables of $\mathcal{T}\{\ \}$ and the free variables of P are disjoint, $P \parallel \mathcal{T}\{\ Q_1, Q_2, \ldots Q_n\ \} \longrightarrow \mathcal{T}\{\ P \parallel Q_1, P \parallel Q_2, \ldots P \parallel Q_n\ \}$.*

For readability of large predicates involving an n-ary killing context, we introduce the notation $\mathcal{T}\{\ Q_i \colon 1 \leq i \leq n\ \}$ as shorthand for $\mathcal{T}\{\ Q_1, Q_2, \ldots, Q_n\ \}$; and $\mathcal{T}\{\ Q_i \colon i \in I\ \}$ for a family of predicates indexed by finite subset of natural numbers I. Killing contexts also satisfy the following property that is necessary for handling the *seq* operator, which interacts subtly with killing contexts.

▶ **Lemma 14.** *Assume that I is a finite subset of natural numbers, P_i and Q_i are predicates, for $i \in I$, and $\mathcal{T}\{\ \}$ is a killing context. There exist killing contexts $\mathcal{T}^0\{\ \}$ and $\mathcal{T}^1\{\ \}$ and sets of natural numbers $J \subseteq I$ and $K \subseteq I$ such that the following derivation holds.*

$$\mathcal{T}\{\ P_i\ ;\ Q_i \colon i \in I\ \} \longrightarrow \mathcal{T}^0\Big\{\ P_j \colon j \in J\ \Big\}\ ;\ \mathcal{T}^1\{\ Q_k \colon k \in K\ \}$$

The following lemma checks that *wen* quantifiers can propagate to the front of a killing context.

▶ **Lemma 15.** *Consider an n-ary killing context $\mathcal{T}\{\ \}$ and predicates such that $x \# P_i$ and either $P_i = \Im x.Q_i$ or $P_i = Q_i$, for $1 \leq i \leq n$. If for some i such that $1 \leq i \leq n$, $P_i = \Im x.Q_i$, then $\mathcal{T}\{\ P_1, P_2, \ldots P_n\ \} \longrightarrow \Im x.\mathcal{T}\{\ Q_1, Q_2, \ldots, Q_n\ \}$.*

4.3 The Splitting Technique for Simulating the Sequent Calculus

The technique called splitting [17, 18] generalises the application of rules in the sequent calculus. In the sequent calculus, any root connective in a sequent can be selected and some rule for that connective can be applied. In this setting, a sequent corresponds to a *shallow context* of the form $\{\ \cdot\ \} \parallel Q$. Splitting proves that a distinguished *principal predicate* P in a shallow context $\{\ P\ \} \parallel Q$ can always be rewritten to a form such that a rule for the root connective of the principal predicate may be

applied. The cases for *times*, *seq*, *new* and *wen* are treated together in a Lemma 16. These operators give rise to *commutative cases*, where rules for these operators can permute with any principal predicate, swapping the order of rules in a proof. *Principal cases* are where the root connective of the principal predicate is directly involved in the bottommost rule of a proof. As with MAV [20], the *principal cases* for *seq* are challenging, demanding Lemma 14. The principal case induced by *medial new* demands Lemma 15. The cases where two nominal quantifiers commute are also interesting, particularly where the case arises due to *equivariance*.

▶ **Lemma 16** (Core Splitting)**.** *The following statements hold.*

- *If* $\vdash (P \otimes Q) \parallel R$*, then there exist predicates* V_i *and* W_i *such that* $\vdash P \parallel V_i$ *and* $\vdash Q \parallel W_i$*, where* $1 \leq i \leq n$*, and n-ary killing context* $\mathcal{T}\{\ \}$ *such that* $R \longrightarrow \mathcal{T}\{ V_1 \parallel W_1, V_2 \parallel W_2, \ldots, V_n \parallel W_n \}$ *and if x appears in* $\mathcal{T}\{\ \}$ *then* $x \# (P \otimes Q)$*.*
- *If* $\vdash (P \,;\, Q) \parallel R$*, then there exist predicates* V_i *and* W_i *such that* $\vdash P \parallel V_i$ *and* $\vdash Q \parallel W_i$*, where* $1 \leq i \leq n$*, and n-ary killing context* $\mathcal{T}\{\ \}$ *such that* $R \longrightarrow \mathcal{T}\{ V_1 \,;\, W_1, V_2 \,;\, W_2, \ldots, V_n \,;\, W_n \}$ *and if x appears in* $\mathcal{T}\{\ \}$ *then* $x \# (P \,;\, Q)$*.*
- *If* $\vdash И x.P \parallel Q$*, then there exist predicates V and W where* $x \# V$ *and* $\vdash P \parallel W$ *and either* $V = W$ *or* $V = Э x.W$*, such that derivation* $Q \longrightarrow V$ *holds.*
- *If* $\vdash Э x.P \parallel Q$*, then there exist predicates V and W where* $x \# V$ *and* $\vdash P \parallel W$ *and either* $V = W$ *or* $V = И x.W$*, such that derivation* $Q \longrightarrow V$ *holds.*

Furthermore, for all $1 \leq i \leq n$*, in the first two cases the size of the proofs* [2] *of* $P \parallel V_i$ *and* $Q \parallel W_i$ *are bounded above by the size of the proofs of* $(P \otimes Q) \parallel R$ *and* $(P \,;\, Q) \parallel R$*. In the third and fourth cases, the size of the proof* $P \parallel W$ *is bounded above by the size of the proofs of* $И x.P \parallel Q$ *and* $Э x.P \parallel Q$*.*

The final three splitting lemmas mainly involve checking commutative cases, which follow a pattern.

▶ **Lemma 17.** *If* $\vdash \exists x.P \parallel Q$*, then there exist predicates* V_i *and values* v_i *such that* $\vdash P\{^{v_i}/_x\} \parallel V_i$*, where* $1 \leq i \leq n$*, and n-ary killing context* $\mathcal{T}\{\ \}$ *such that* $Q \longrightarrow \mathcal{T}\{ V_1, V_2, \ldots, V_n \}$ *and if y appears in* $\mathcal{T}\{\ \}$ *then* $y \# (\exists x.P)$*.*

▶ **Lemma 18.** *The following statements hold, for any atom* α*, where if x appears in* $\mathcal{T}\{\ \}$ *then* $x \# \alpha$*.*

- *If* $\vdash \overline{\alpha} \parallel Q$*, then there exist n-ary killing context* $\mathcal{T}\{\ \}$ *such that* $Q \longrightarrow \mathcal{T}\{ \alpha, \alpha, \ldots, \alpha \}$*.*
- *If* $\vdash \alpha \parallel Q$*, then there exist n-ary killing context* $\mathcal{T}\{\ \}$ *such that* $Q \longrightarrow \mathcal{T}\{ \overline{\alpha}, \overline{\alpha}, \ldots, \overline{\alpha} \}$*.*

▶ **Lemma 19.** *If* $\vdash (P \oplus Q) \parallel R$*, then there exist predicates* W_i *such that either* $\vdash P \parallel W_i$ *or* $\vdash Q \parallel W_i$ *where* $1 \leq i \leq n$*, and n-ary killing context* $\mathcal{T}\{\ \}$ *such that* $R \longrightarrow \mathcal{T}\{ W_1, W_2, \ldots, W_n \}$ *and if x appears in* $\mathcal{T}\{\ \}$ *then* $x \# (P \oplus Q)$*.*

4.4 Context Reduction and the Admissibility of Co-rules

Splitting is always performed in a *shallow context*, i.e. a hole directly inside a parallel composition. Context reduction extends rules simulated by splitting to any context.

▶ **Lemma 20** (Context reduction)**.** *If* $\vdash P\sigma \parallel R$ *yields that* $\vdash Q\sigma \parallel R$*, for any predicate R and substitution of terms for variables* σ*, then* $\vdash C\{ P \}$ *yields* $\vdash C\{ Q \}$*, for any context* $C\{\ \}$*.*

The subtlety of context reduction is to accommodate *plus* and *some* by the following stronger induction invariant: If $\vdash C\{ T \}$, then there exist predicates U_i and substitutions σ_i such that $\vdash T\sigma_i \parallel U_i$, for $1 \leq i \leq n$; and n-ary killing context $\mathcal{T}\{\ \}$ such that for any predicate V there exist W_i such that either $W_i = V\sigma_i \parallel U_i$ or $W_i = \mathbb{1}$, for $1 \leq i \leq n$, and the following holds: $C\{ V \} \longrightarrow \mathcal{T}\{ W_1, W_2, \ldots, W_n \}$.

[2] For the multiset measure of the size of a proof see `http://iit.iit.tuiasi.ro/TR/reports/fml1502.pdf`.

For every rule there is a *co-rule*, where for rule $P \longrightarrow Q$, the co-rule is of the form $\overline{Q} \longrightarrow \overline{P}$. For example *close* has co-rule $C\{\ \Im x.(P \otimes Q)\ \} \longrightarrow C\{\ \Im x.P \otimes \mathrel{И} x.Q\ \}$ and *extrude1* has co-rule if $x \# Q$ then $C\{\ \exists x.(P \otimes Q)\ \} \longrightarrow C\{\ \exists x.P \otimes Q\ \}$. The following eight lemmas establish that a co-rule is admissible in MAV1. In each case, the proof proceeds by applying splitting in a shallow context, forming a new proof, and finally applying Lemma 20.

▶ **Lemma 21** (Co-close). *If* $\vdash C\{\ \Im x.P \otimes \mathrel{И} x.Q\ \}$ *holds then* $\vdash C\{\ \Im x.(P \otimes Q)\ \}$ *holds.*

▶ **Lemma 22** (Co-tidy name). *If* $\vdash C\{\ \Im x.\mathrm{I}\ \}$ *holds then* $\vdash C\{\ \mathrm{I}\ \}$ *holds.*

▶ **Lemma 23** (Co-extrude1). *If* $x \# Q$ *and* $\vdash C\{\ \exists x.P \otimes Q\ \}$ *holds then* $\vdash C\{\ \exists x.(P \otimes Q)\ \}$ *holds.*

▶ **Lemma 24** (Co-tidy1). *If* $\vdash C\{\ \exists x.\mathrm{I}\ \}$ *holds then* $\vdash C\{\ \mathrm{I}\ \}$ *holds.*

The above four lemmas are particular to MAV1. The proofs for the four lemmas below are similar to the corresponding cases in MAV [20].

▶ **Lemma 25** (Co-external). *If* $\vdash C\{\ P \otimes (Q \oplus R)\ \}$ *holds then* $\vdash C\{\ (P \otimes Q) \oplus (P \otimes R)\ \}$ *holds.*

▶ **Lemma 26** (Co-tidy). *If* $\vdash C\{\ \mathrm{I} \oplus \mathrm{I}\ \}$ *holds, then* $\vdash C\{\ \mathrm{I}\ \}$ *holds.*

▶ **Lemma 27** (Co-sequence). *If* $\vdash C\{\ (P\ ;\ Q) \otimes (R\ ;\ S)\ \}$ *holds then* $\vdash C\{\ (P \otimes R)\ ;\ (Q \otimes S)\ \}$ *holds.*

▶ **Lemma 28** (Atomic co-interaction). *If* $\vdash C\{\ \alpha \otimes \overline{\alpha}\ \}$ *holds then* $\vdash C\{\ \mathrm{I}\ \}$ *holds.*

The Proof of Theorem 4. The main result of this paper follows by induction on the structure of P in a predicate of the form $\vdash C\left\{\ P \otimes \overline{P}\ \right\}$, by applying the above eight co-rule elimination lemmas and also Lemma 10 in the cases for *with* and *plus*, and Lemma 11 in the cases for *all* and *some*.

Co-rules are interesting in their own right, since derivations extended with all co-rules coincide with provable linear implications. Suppose that SMAV1 is the system MAV1 extended with all co-rules. The following corollary is a consequence of Theorem 4.

▶ **Corollary 29.** $\vdash P \multimap Q$ *in MAV1 if and only if* $Q \longrightarrow P$ *in SMAV1.*

An advantage of defining linear implication using provability, is that MAV1 is *analytic* [8]; hence, with some care taken for existential quantifiers [5, 23], each predicate gives rise to finitely many derivations up-to congruence. Consequently, proof search is decidable.

▶ **Proposition 30.** *It is decidable whether a predicate in MAV1 is provable.*

5 Conclusion

This paper makes two significant contributions: a novel de Morgan dual pair of nominal quantifiers and the first direct cut elimination result for first-order quantifiers in the calculus of structures. As a consequence of cut-elimination (Theorem 4), we obtain the first consistent proof system that features both non-commutative operator *seq* and first-order quantifiers. The novelty of the nominal quantifiers *new* and *wen* is in how they distribute over positive and negative operators. This greater control of bookkeeping of names enables private names to be modelled in direct embeddings of processes as predicates in MAV1.

This paper continues a line of work on logical systems defined using the calculus of structures with applications to modelling processes [7, 12, 20]. Our approach is distinct from related work on nominal logic [2, 13, 38] where processes are terms, rather than predicates, and an operational semantics is given either as an inductive definition or a logical theory. The related approach is capable of encoding

observational preorders and bisimulations, but has the drawback that implication cannot be directly used to compare processes. Our approach is also distinct from the proofs-as-processes Curry-Howard inspired approach to session types [9, 42]. Instead, we adopt a processes-as-predicates approach, setting up a discussion on the relationship between linear implication in MAV1 and observational preorders. Amongst the consequences of cut elimination (Theorem 4) is that linear implication defines a branching-time precongruence over processes that fully respects causality.

Acknowledgements The first two authors receive support from MOE Tier 2 grant MOE2014-T2-2-076. The second author receives support from NTU Start Up grant M4081190.020. The first, third and fourth authors are supported by ANCS grant number PN-II-ID-PCE-2011-3-0919. We are grateful to Catuscia Palamidessi and the CONCUR panel for their thorough analysis of a draft.

References

1 Luca Aceto and Matthew Hennessy. Adding action refinement to a finite process algebra. *Information and Computation*, 115(2):179–247, 1994. doi:10.1006/inco.1994.1096.

2 Andrei Alexandru and Gabriel Ciobanu. Nominal techniques for πI-calculus. *Romanian Journal of Information Science and Technology*, 16(4):261–286, 2013.

3 Jean-Marc Andreoli. Logic programming with focusing proofs in linear logic. *Journal of Logic and Computation*, 2(3):297–347, 1992. doi:10.1093/logcom/2.3.297.

4 Marek Bednarczyk. Hereditary history preserving bisimulations. Technical report, Polish Academy of Sciences, Gdańsk, 1991.

5 Kai Brünnler. *Deep inference and symmetry in classical proofs*. PhD thesis, TU Dresden, 2003.

6 Kai Brünnler. Locality for classical logic. *Notre Dame J. Form. Log.*, 47(4):557–580, 2006.

7 Paola Bruscoli. A purely logical account of sequentiality in proof search. In *ICLP*, volume 2401 of *LNCS*, pages 302–316. Springer, 2002. doi:10.1007/3-540-45619-8_21.

8 Paola Bruscoli and Alessio Guglielmi. On the proof complexity of deep inference. *ACM Transactions on Computational Logic (TOCL)*, 10(2), 2009. doi:10.1145/1462179.1462186.

9 Luís Caires and Frank Pfenning. Session types as intuitionistic linear propositions. In *CONCUR 2010*, pages 222–236. Springer, 2010. doi:10.1007/978-3-642-15375-4_16.

10 Kaustuv Chaudhuri, Nicolas Guenot, and Lutz Straßburger. The focused calculus of structures. In *EACSL*, volume 12, pages 159–173, 2011. doi:10.4230/LIPIcs.CSL.2011.159.

11 Gabriel Ciobanu and Ross Horne. Non-interleaving operational semantics for geographically replicated databases. In *SYNASC 2013*, pages 440–447, 2013. doi:10.1109/SYNASC.2013.64.

12 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*, 2015.

13 Murdoch J. Gabbay. The π-calculus in FM. In *Thirty Five Years of Automating Mathematics*, pages 247–269. Springer, 2003. doi:10.1007/978-94-017-0253-9_10.

14 Andrew Gacek, Dale Miller, and Gopalan Nadathur. Nominal abstraction. *Information and Computation*, 209(1):48–73, 2011. doi:10.1016/j.ic.2010.09.004.

15 Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50(1):1–112, 1987. doi:10.1016/0304-3975(87)90045-4.

16 Alessio Guglielmi. Re:encoding pi calculus in calculus of structures. post on public mailing list http://permalink.gmane.org/gmane.science.mathematics.frogs/161, 2004.

17 Alessio Guglielmi. A system of interaction and structure. *ACM Transactions on Computational Logic*, 8(1), 2007. doi:10.1145/1182613.1182614.

18 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. doi:10.1017/S096012951100003X.

19 Kohei Honda et al. Scribbling interactions with a formal foundation. In *ICDCIT 2011*, volume 6536 of *LNCS*, pages 55–75. Springer, 2011. doi:10.1007/978-3-642-19056-8_4.

20 Ross Horne. The consistency and complexity of multiplicative additive system virtual. *Sci. Ann. Comp. Sci.*, 25(2):245–316, 2015. URL: http://dx.doi.org/10.7561/SACS.2015.2.245.

21 Naoki Kobayashi and Akinori Yonezawa. ACL – a concurrent linear logic programming paradigm. In *ILPS'93*, pages 279–294. MIT Press, 1993.

22 Leslie Lamport. Time, clocks, and the ordering of events in a distributed system. *Communications of the ACM*, 21(7):558–565, 1978. doi:10.1145/359545.359563.

23 Patrick Lincoln and Natarajan Shankar. Proof search in first-order linear logic and other cut-free sequent calculi. In *LICS'94*, pages 282–291. IEEE, 1994. doi:10.1109/LICS.1994.316061.

24 José Meseguer. Conditional rewriting logic as a unified model of concurrency. *Theoretical Computer Science*, 96(1):73–155, 1992. doi:10.1016/0304-3975(92)90182-F.

25 Dale Miller and Alwen Tiu. A proof theory for generic judgements. *ACM Transactions on Computational Logic (TOCL)*, 6(4):749–783, 2005. doi:10.1145/1094622.1094628.

26 Robin Milner, Joachim Parrow, and David Walker. A calculus of mobile processes, I and II. *Information and Computation*, 100(1):1–77, 1992. doi:10.1016/0890-5401(92)90008-4.

27 Peter O'Hearn and David Pym. The logic of bunched implications. *Bulletin of Symbolic Logic*, 5(2):215–244, 1999. doi:10.2307/421090.

28 Andrew Pitts. Nominal logic, a first order theory of names and binding. *Information and Computation*, 186(2), 2003. doi:10.1016/S0890-5401(03)00138-X.

29 Vaughan Pratt. Modelling concurrency with partial orders. *International Journal of Parallel Programming*, 15(1):33–71, 1986. doi:10.1007/BF01379149.

30 Christian Retoré. Pomset logic: A non-commutative extension of classical linear logic. In *TLCA'97*, volume 1210 of *LNCS*, pages 300–318. Springer, 1997. doi:10.1007/3-540-62688-3_43.

31 Luca Roversi. A deep inference system with a self-dual binder which is complete for linear lambda calculus. *J. of Log. and Comp.*, 26(2):677–698, 2016. doi:10.1093/logcom/exu033.

32 Davide Sangiorgi. π-calculus, internal mobility, and agent-passing calculi. *Theoretical Computer Science*, 167(1):235–274, 1996. doi:10.1016/0304-3975(96)00075-8.

33 Vladimiro Sassone, Mogens Nielsen, and Glynn Winskel. Models for concurrency: towards a classification. *Th. Comp. Sci.*, 170(1-2):297–348, 1996. doi:10.1016/S0304-3975(96)80710-9.

34 Lutz Straßburger. *Linear logic and noncommutativity in the calculus of structures*. PhD thesis, TU Dresden, 2003.

35 Lutz Straßburger. Some observations on the proof theory of second order propositional multiplicative linear logic. In *TLCA 2009*, volume 5608 of *LNCS*, pages 309–324. Springer, 2009. doi:10.1007/978-3-642-02273-9_23.

36 Lutz Straßburger and Alessio Guglielmi. A system of interaction and structure IV: the exponentials and decomposition. *TOCL*, 12(4):23, 2011. doi:10.1145/1970398.1970399.

37 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. doi:10.2168/LMCS-2(2:4)2006.

38 Alwen Tiu and Dale Miller. Proof search specifications of bisimulation and modal logics for the π-calculus. *TOCL*, 11(2):13, 2010. doi:10.1145/1656242.1656248.

39 Rob van Glabbeek. The linear time-branching time spectrum (extended abstract). In *CONCUR '90*, volume 458 of *LNCS*, pages 278–297. Springer, 1990. doi:10.1007/BFb0039066.

40 Rob van Glabbeek. Structure preserving bisimilarity, supporting an operational Petri net semantics of CCSP. In *Correct System Design*, volume 9360 of *LNCS*, pages 99–130. Springer, 2015. doi:10.1007/978-3-319-23506-6_9.

41 Rob van Glabbeek and Ursula Goltz. Refinement of actions and equivalence notions for concurrent systems. *Acta Informatica*, 37(4-5):229–327, 2001. doi:10.1007/s002360000041.

42 Philip Wadler. Propositions as sessions. *J. of Fun. Prog.*, 24(2-3):384–418, 2014. doi:10.1145/2364527.2364568.