Ambient Logic

Luca Cardelli and Andrew D. Gordon

Microsoft Research

Abstract. The Ambient Calculus is a process calculus where processes may reside within a hierarchy of locations. The purpose of this calculus is to study mobility; to this end, processes can move through the location hierarchy and modify it. Therefore, mobility is seen as the change of spatial configurations over time. In order to describe properties of mobile computations we devise a modal logic, solidly based on the Ambient Calculus, that can talk about space as well as time. We introduce logical operators that can be used to make assertions about locations and their names, and we study their properties.

1 Introduction

In the course of our ongoing work on mobility [12,10,26], we have often struggled to express precisely certain properties of mobile computations. Informally, these are properties such as "the agent has gone away", "eventually the agent crosses the firewall", "every agent always carries a suitcase", "somewhere there is a virus", or "there is always at most one agent called n here". There are several conceivable ways of formalizing these assertions. It is possible to express some of them in terms of equations [26], but this is sometimes difficult or unnatural. It is easier to express some of them as properties of computational traces, but this is very low-level.

Modal logics (particularly, temporal logics) have emerged in many domains as a good compromise between expressiveness and abstraction. In addition, many modal logics support useful computational applications, such as model checking. In our context, it makes sense to talk about properties that hold at particular locations, and it becomes natural to consider *spatial modalities* for properties that hold at a certain location, at some location or at every location. For example, we have the following correspondence between spatial constructs in the Ambient Calculus [12] and certain formulas:

Processes		*Formulas*	
$\mathbf{0}$	(void)	$\mathbf{0}$	(there is nothing here)
$n[P]$	(location)	$n[\mathcal{A}]$	(there is one thing here)
$P \mid Q$	(composition)	$\mathcal{A} \mid \mathcal{B}$	(there are two things here)

We have a logical constant $\mathbf{0}$ that is satisfied by the process $\mathbf{0}$ representing void. We have logical propositions of the form $n[\mathcal{A}]$ (meaning that $\mathcal{A}$ holds at location n) that are satisfied by processes of the form $n[P]$ (meaning that process P is located at n) provided that P satisfies $\mathcal{A}$. We have logical propositions of the form $\mathcal{A}' \mid \mathcal{A}''$ (meaning that $\mathcal{A}'$ and $\mathcal{A}''$ hold contiguously) which are satisfied by contiguous processes of the form $P' \mid P''$ if P' satisfies

$\mathcal{A}'$ and P'' satisfies $\mathcal{A}''$, or vice versa.

These spatial modalities have an intensional flavor that distinguishes our logic from other modal logics for concurrency. Previous work in the area concentrates on properties that are invariant up to strong equivalences such as bisimulation [27,19], while our properties are invariant only up to simple spatial rearrangements. Some of our techniques can be usefully applied to other process calculi, even ones that do not have locations [5].

The π-calculus notion of name restriction [30], initially intended to represent hidden communication channels, has been used also to represent hidden encryption keys [2] and as the basis for definitions of secrecy [2, 11]. In the context of the Ambient Calculus [12], name restriction can be used to represent hidden locations and (by extrapolating [11] and [10]) secret locations. In general, we would like to have process calculi where we can represent protocols for creating shared encryption keys and secret locations; name restriction seems crucial to all this.

In π-calculus notation, $(\nu n)P$ is a restriction of the name n in the process P, meaning that n is not currently known outside the scope of P. The prefix (νn) is more a bookkeeping device than a barrier. It is quite possible for P to communicate n to some external process; then the restriction (νn) must be formally pushed outwards to encompass the new scope of n and maintain the scoping invariant; this procedure is called *name extrusion*. Processes are considered equivalent up to extrusion; that is, extrusion is not regarded as a computational step. Conversely, when a name is forgotten in part of a process, the scope of (νn) may be restricted; this is called name *intrusion*. Manipulation of (νn) prefixes includes, in particular, renaming and swapping of prefixes, so that there is no obvious way of talking about "the first restricted name" or any particular restricted name of a process.

With our spatial modalities, we can describe detailed properties of processes. If we now consider restriction, what does it mean to describe properties of restricted names? We would like to be able to say, for example, "a shared key is established between locations a and b", or "a secret location is created that only a and b can access". In a protocol that establishes such shared secrets, the secrets are typically represented by restricted names. The problem is that there is no obvious way to talk about such restricted names in the specification of the protocol. We might be tempted to use ordinary existential quantification, and say "there exists a name shared between locations a and b". But this is not good enough, because we want that name to be fresh and unknown to other locations or potential attackers.

Therefore, we want a new form of quantification that can be read as "in the process there exists a restricted name which we shall call x, and such that $\mathcal{A}$", where x is a variable that ranges over names, and $\mathcal{A}$ is some property that may involve x. Let us indicate this quantifier as $\mathsf{H}x.\mathcal{A}$; this formula is meant to correspond somehow to a process of the form $(\nu n)P$ where x denotes n. However, since (νn) can float, the matching of $\mathsf{H}x$ to any particular (νn) is not obvious.

This means that the logical rules of our tentative $\mathsf{H}x.\mathcal{A}$ quantifier are going to be fairly complex, or at least unfamiliar. We have approached this complexity by splitting $\mathsf{H}x.\mathcal{A}$ into two operators; one for quantifying over fresh names, and one for mentioning restricted names. The first operator is the Gabbay-Pitts quantifier, $\mathsf{N}x.\mathcal{A}$, adapted to our context: it quantifies over all names that do not occur free either in the formula $\mathcal{A}$ or in the described process. The second is a binary operator (not a quantifier) called *revelation*, $n@\mathcal{A}$, which

means that it is possible to reveal a restricted name as the given name n, and then assert $\mathcal{A}$. (Revelation fails to hold if it would lead to a name clash in the process.)

We investigate the properties of $n \circledR \mathcal{A}$ and $\forall x.\mathcal{A}$ separately. We combine them to define $\exists x.\mathcal{A}$ as $\forall x.x \circledR \mathcal{A}$, and then we study the derived properties of $\exists x.\mathcal{A}$.

This paper is a updated version of both [13] and [14], bringing together those conference papers into a better integrated and self-contained whole. In Section 2 we give an overview of the Ambient Calculus. In Section 3 we introduce our Ambient Logic and its satisfaction semantics. In Section 4 we define logical sequents, and we derive a number of logical truths for all logical connectives except revelation and hiding. In Section 5 we study the logical properties of revelation and hiding, and explain the difficulties in defining a hidden-name quantifier. To that end, in Section 6 we study a fresh-name quantifier. In Section 7 we are then able to define a proper hidden-name quantifier, and we investigate its properties. In Section 8 we compare our logic to other logics, and in particular to Intuitionistic Linear Logic. In Section 9 we discuss related work.

2 The Ambient Calculus

2.1 Ambients

We summarize a modified version of the basic Ambient Calculus of [12]. The changes consist in strengthening the definition of structural congruence so that it characterizes the intended equivalence of spatial configurations.

The following table summarizes the syntax of processes. We have separated the process constructs into *spatial* and *temporal*. This is similar to the distinction between static and dynamic constructs in CCS [29]. This paper focuses on the spatial constructs; the temporal constructs and the dynamic behavior are necessary but secondary, for our purposes.

2.2 Ambients

Processes

P,Q,R ::=	processes		
$(\nu n)P$	restriction		
$\mathbf{0}$	void		
$P \mid Q$	composition		spatial
$!P$	replication		
$M[P]$	ambient		
$M.P$	capability action		
$(n).P$	input action		temporal
$\langle M \rangle$	output action		

M ::=	capabilities
n	name
in M	can enter into M
out M	can exit out of M
open M	can open M
ε	null
$M.M'$	path

The set of free names of a process P, written $fn(P)$ is defined as follows, where the only binders are restriction and the input action.

Free names

$fn((\nu n)P) \triangleq fn(P) - \{n\}$
$fn(\mathbf{0}) \triangleq \emptyset$
$fn(P \mid Q) \triangleq fn(P) \cup fn(Q)$
$fn(!P) \triangleq fn(P)$
$fn(M[P]) \triangleq fn(M) \cup fn(P)$
$fn(M.P) \triangleq fn(M) \cup fn(P)$
$fn((n).P) \triangleq fn(P) - \{n\}$
$fn(\langle M \rangle) \triangleq fn(M)$

$fn(n) \triangleq \{n\}$
$fn(in\ M) \triangleq fn(M)$
$fn(out\ M) \triangleq fn(M)$
$fn(open\ M) \triangleq fn(M)$
$fn(\varepsilon) \triangleq \emptyset$
$fn(M.M') \triangleq fn(M) \cup fn(M')$

We write $P\{n \leftarrow M\}$ for the substitution of the capability M for each free occurrence of the name n in the process P. Similarly for $M\{n \leftarrow M'\}$. We identify processes up to renaming of bound names, that is, we assume the identities $(\nu n)P = (\nu m)P\{n \leftarrow m\}$ and $(n).P = (m).P\{n \leftarrow m\}$, where, in both equations, $m \notin fn(P)$.

We use some syntactic conventions. We use parentheses for precedence. The process $\mathbf{0}$ is often omitted in the contexts $n[\mathbf{0}]$ and $M.\mathbf{0}$, yielding $n[]$ and M. Composition has the weakest binding power, so that the expression $(\nu n)P \mid Q$ is read $((\nu n)P) \mid Q$, the expression $!P \mid Q$ is read $(!P) \mid Q$, the expression $M.P \mid Q$ is read $(M.P) \mid Q$, and the expression $(n).P \mid Q$ is read $((n).P) \mid Q$.

Structural congruence is a relation between processes used as an aid in the definition of reduction. With respect to [12], the structural rules for replication have been refined.

Structural Congruence

$P \equiv P$	(Struct Refl)
$P \equiv Q \Rightarrow Q \equiv P$	(Struct Symm)
$P \equiv Q, Q \equiv R \Rightarrow P \equiv R$	(Struct Trans)
$P \equiv Q \Rightarrow (\nu n)P \equiv (\nu n)Q$	(Struct Res)
$P \equiv Q \Rightarrow P \mid R \equiv Q \mid R$	(Struct Par)
$P \equiv Q \Rightarrow !P \equiv !Q$	(Struct Repl)
$P \equiv Q \Rightarrow n[P] \equiv n[Q]$	(Struct Amb)
$P \equiv Q \Rightarrow M.P \equiv M.Q$	(Struct Action)
$P \equiv Q \Rightarrow (n).P \equiv (n).Q$	(Struct Input)

$\varepsilon.P \equiv P$	(Struct ε)
$(M.M').P \equiv M.M'.P$	(Struct .)
$(\nu n)(\nu m)P \equiv (\nu m)(\nu n)P$	(Struct Res Res)
$(\nu n)\mathbf{0} \equiv \mathbf{0}$	(Struct Res Zero)
$(\nu n)(P \mid Q) \equiv P \mid (\nu n)Q \quad \text{if } n \notin fn(P)$	(Struct Res Par)
$(\nu n)(m[P]) \equiv m[(\nu n)P] \quad \text{if } n \neq m$	(Struct Res Amb)
$P \mid \mathbf{0} \equiv P$	(Struct Par Zero)
$P \mid Q \equiv Q \mid P$	(Struct Par Comm)
$(P \mid Q) \mid R \equiv P \mid (Q \mid R)$	(Struct Par Assoc)
$!\mathbf{0} \equiv \mathbf{0}$	(Struct Repl Zero)
$!(P \mid Q) \equiv !P \mid !Q$	(Struct Repl Par)
$!P \equiv P \mid !P$	(Struct Repl Copy)
$!P \equiv !!P$	(Struct Repl Repl)

The reduction relation, defined in the following table, describes the dynamic behavior of ambients. In particular, the rules (Red In), (Red Out) and (Red Open) represent mobility, while (Red Comm) represents local communication (see [12] for an extended discussion). For example, the process $a[p[\mathit{out}\ a.\ \mathit{in}\ b.\ \langle m\rangle]] \mid b[\mathit{open}\ p.\ (n).\ n[]]$ represents a packet p that travels out of host a and into host b, where it is opened, and its contents m are read and used to create a new ambient. The process reduces in four steps (illustrating each of the four reduction rules) to the residual process $a[] \mid b[m[]]$.

Reduction

$n[\mathit{in}\ m.\ P \mid Q] \mid m[R] \longrightarrow m[n[P \mid Q] \mid R]$	(Red In)
$m[n[\mathit{out}\ m.\ P \mid Q] \mid R] \longrightarrow n[P \mid Q] \mid m[R]$	(Red Out)
$\mathit{open}\ n.\ P \mid n[Q] \longrightarrow P \mid Q$	(Red Open)
$(n).P \mid \langle M\rangle \longrightarrow P\{n \leftarrow M\}$	(Red Comm)
$P \longrightarrow Q \ \Rightarrow\ (\nu n)P \longrightarrow (\nu n)Q$	(Red Res)
$P \longrightarrow Q \ \Rightarrow\ P \mid R \longrightarrow Q \mid R$	(Red Par)
$P \longrightarrow Q \ \Rightarrow\ n[P] \longrightarrow n[Q]$	(Red Amb)
$P' \equiv P,\ P \longrightarrow Q,\ Q \equiv Q' \ \Rightarrow\ P' \longrightarrow Q'$	(Red ≡)
$\longrightarrow^{*}$	reflexive and transitive closure of $\longrightarrow$

2-1 Lemmas

(1) $P \equiv Q \ \Rightarrow\ fn(P) = fn(Q)$

(2) $P \mid Q \equiv \mathbf{0}$ iff $P \equiv \mathbf{0}$ and $Q \equiv \mathbf{0}$.

(3) $n[P] \not\equiv \mathbf{0}$.

(4) $n[P] \equiv Q \mid R$ iff either $Q \equiv n[P]$ and $R \equiv \mathbf{0}$, or $Q \equiv \mathbf{0}$ and $R \equiv n[P]$.

(5) $m[P] \equiv n[Q]$ iff $m = n$ and $P \equiv Q$.

(6) $m[P] \mid n[Q] \equiv m'[P'] \mid n'[Q']$ iff:
either $m = m'$, $n = n'$, $P \equiv P'$, $Q \equiv Q'$,
or $m = n'$, $n = m'$, $P \equiv Q'$, $Q \equiv P'$.
(7) $(\nu n)P \equiv \mathbf{0} \;\Rightarrow\; P \equiv \mathbf{0}$
(8) $(\nu n)P \equiv m[Q] \;\Rightarrow\; \exists R \in \Pi.\ P \equiv m[R] \wedge Q \equiv (\nu n)R$ (for $n \neq m$)
(9) $(\nu n)P \equiv Q' \mid Q'' \;\Rightarrow\; \exists R', R'' \in \Pi.\ P \equiv R' \mid R'' \wedge Q' \equiv (\nu n)R' \wedge Q'' \equiv (\nu n)R''$
□

See [17] for proofs of these lemmas.

Remark. It is not true that $(\nu n)P \equiv (\nu n)Q$ implies $P \equiv Q$. Take $P = n[]$ and $Q = (\nu n)n[]$; then $(\nu n)n[] \equiv (\nu n)(\nu n)n[]$ but $n[] \not\equiv (\nu n)n[]$. □

3 The Logic

In a modal logic, the truth of a formula is relative to a state (or world). In our case, the truth of a *space-time* modal formula is relative to the *here and now*. Each formula talks about the current time, that is, the current state of execution, and the current place, that is, the current location. For example, the formula $n[\mathbf{0}]$ is read: *there is here and now an empty location called n*. The operator $n[\mathcal{A}]$ represents a single step in space, allowing us to talk about the place one step down into *n*. Another operator, $\diamond\!\!\!\!+\mathcal{A}$, allows us to talk about an arbitrary number of steps in space; this is akin to the temporal eventuality operator, $\diamond\mathcal{A}$.

3.1 Logical Formulas

The syntax of logical formulas is summarized below. This is a modal predicate logic with classical negation. As usual, many standard connectives are interdefinable; we take **T**, $\neg$, $\vee$, $\diamond$, $\forall$ as primitive, and **F**, $\Rightarrow$, $\wedge$, $\Box$, $\exists$ as derived.

The meaning of the formulas will be given shortly in terms of a satisfaction relation. Informally, the first three formulas (true, negation, disjunction) give propositional logic. The next five (void, composition and its adjunct, location and its adjunct) describe tree-like structures of locations. Revelation and its adjunct are discussed in detail later. The two spatial and temporal modalities make assertions about states that may happen "further away" in space or time respectively. Quantified variables range only over names: these variables may appear in the location and revelation constructs, and their adjuncts.

Logical Formulas

η, μ	a name n or a variable x

$\mathcal{A}, \mathcal{B}, C$ ::=

$\mathbf{T}$	true
$\neg\mathcal{A}$	negation
$\mathcal{A} \vee \mathcal{B}$	disjunction
$\mathbf{0}$	void
$\mathcal{A} \mid \mathcal{B}$	composition
$\mathcal{A} \triangleright \mathcal{B}$	guarantee
$\eta[\mathcal{A}]$	location
$\mathcal{A}@\eta$	placement
$\eta \circledR \mathcal{A}$	revelation
$\mathcal{A} \oslash \eta$	hiding
$\Diamond\mathcal{A}$	sometime modality
$\varkappa\mathcal{A}$	somewhere modality
$\forall x.\mathcal{A}$	universal quantification

Free Names and Free Variables

$fn(n)$	$\triangleq$	$\{n\}$	$fv(n)$	$\triangleq$	$\emptyset$
$fn(x)$	$\triangleq$	$\emptyset$	$fv(x)$	$\triangleq$	$\{x\}$
$fn(\mathbf{T})$	$\triangleq$	$\emptyset$	$fv(\mathbf{T})$	$\triangleq$	$\emptyset$
$fn(\neg\mathcal{A})$	$\triangleq$	$fn(\mathcal{A})$	$fv(\neg\mathcal{A})$	$\triangleq$	$fv(\mathcal{A})$
$fn(\mathcal{A} \vee \mathcal{B})$	$\triangleq$	$fn(\mathcal{A}) \cup fn(\mathcal{B})$	$fv(\mathcal{A} \vee \mathcal{B})$	$\triangleq$	$fv(\mathcal{A}) \cup fv(\mathcal{B})$
$fn(\mathbf{0})$	$\triangleq$	$\emptyset$	$fv(\mathbf{0})$	$\triangleq$	$\emptyset$
$fn(\mathcal{A} \mid \mathcal{B})$	$\triangleq$	$fn(\mathcal{A}) \cup fn(\mathcal{B})$	$fv(\mathcal{A} \mid \mathcal{B})$	$\triangleq$	$fv(\mathcal{A}) \cup fv(\mathcal{B})$
$fn(\mathcal{A} \triangleright \mathcal{B})$	$\triangleq$	$fn(\mathcal{A}) \cup fn(\mathcal{B})$	$fv(\mathcal{A} \triangleright \mathcal{B})$	$\triangleq$	$fv(\mathcal{A}) \cup fv(\mathcal{B})$
$fn(\eta[\mathcal{A}])$	$\triangleq$	$fn(\eta) \cup fn(\mathcal{A})$	$fv(\eta[\mathcal{A}])$	$\triangleq$	$fv(\eta) \cup fv(\mathcal{A})$
$fn(\mathcal{A}@\eta)$	$\triangleq$	$fn(\mathcal{A}) \cup fn(\eta)$	$fv(\mathcal{A}@\eta)$	$\triangleq$	$fv(\mathcal{A}) \cup fv(\eta)$
$fn(\eta \circledR \mathcal{A})$	$\triangleq$	$fn(\eta) \cup fn(\mathcal{A})$	$fv(\eta \circledR \mathcal{A})$	$\triangleq$	$fv(\eta) \cup fv(\mathcal{A})$
$fn(\mathcal{A} \oslash \eta)$	$\triangleq$	$fn(\mathcal{A}) \cup fn(\eta)$	$fv(\mathcal{A} \oslash \eta)$	$\triangleq$	$fv(\mathcal{A}) \cup fv(\eta)$
$fn(\Diamond\mathcal{A})$	$\triangleq$	$fn(\mathcal{A})$	$fv(\Diamond\mathcal{A})$	$\triangleq$	$fv(\mathcal{A})$
$fn(\varkappa\mathcal{A})$	$\triangleq$	$fn(\mathcal{A})$	$fv(\varkappa\mathcal{A})$	$\triangleq$	$fv(\mathcal{A})$
$fn(\forall x.\mathcal{A})$	$\triangleq$	$fn(\mathcal{A})$	$fv(\forall x.\mathcal{A})$	$\triangleq$	$fv(\mathcal{A})\text{-}\{x\}$
$fn(\mathcal{A}_1,\ldots,\mathcal{A}_k)$	$\triangleq$	$fn(\mathcal{A}_1) \cup \ldots \cup fn(\mathcal{A}_k)$	$fv(\mathcal{A}_1,\ldots,\mathcal{A}_k)$	$\triangleq$	$fv(\mathcal{A}_1) \cup \ldots \cup fv(\mathcal{A}_k)$

A formula $\mathcal{A}$ is closed if $fv(\mathcal{A}) = \emptyset$. Substitution $\mathcal{A}\{\eta \leftarrow \mu\}$ of a name or variable μ for another name or variable η in a formula $\mathcal{A}$, is defined in the usual way. We identify formulas up to renaming of bound variables, that is, we assume the identity $\forall x.\mathcal{A} = \forall y.\mathcal{A}\{x \leftarrow y\}$, where $y \notin fv(\mathcal{A})$. We often write $\eta[]$ for $\eta[\mathbf{0}]$, $\mathcal{A}^{\mathbf{F}}$ for $\mathcal{A} \triangleright \mathbf{F}$, and $\mathcal{A}^{\neg}$ for $\neg\mathcal{A}$.

3.2 Satisfaction

The satisfaction relation $P \models \mathcal{A}$ means that the process P satisfies the closed formula $\mathcal{A}$. The definition of satisfaction is based heavily on the structural congruence relation. The satisfaction relation is defined inductively in the following tables, where Π is the sort of processes, Φ is the sort of formulas, ϑ is the sort of variables, and Λ is the sort of names. We use similar syntax for logical connectives at the meta-level and object-level, but this is unambiguous.

The meaning of the temporal modality is given by reductions in the operational semantics of the Ambient Calculus. For the spatial modality, we need the following definitions. The relation $P \downarrow P'$ indicates that P contains P' within exactly one level of nesting. Then, $P \downarrow^* P'$ is the reflexive and transitive closure of the previous relation, indicating that P contains P' at some nesting level. Note that P' constitutes the entire contents of an enclosed ambient.

$$P \downarrow P' \quad \text{iff} \quad \exists n, P''.\ P \equiv n[P'] \mid P''$$

$\downarrow^*$ is the reflexive and transitive closure of $\downarrow$

Satisfaction

$\forall P \in \Pi.$	$P \models \mathbf{T}$		
$\forall P \in \Pi, \mathcal{A} \in \Phi.$	$P \models \neg\mathcal{A}$	$\triangleq$	$\neg\, P \models \mathcal{A}$
$\forall P \in \Pi, \mathcal{A},\mathcal{B} \in \Phi.$	$P \models \mathcal{A} \vee \mathcal{B}$	$\triangleq$	$P \models \mathcal{A} \vee P \models \mathcal{B}$
$\forall P \in \Pi.$	$P \models \mathbf{0}$	$\triangleq$	$P \equiv \mathbf{0}$
$\forall P \in \Pi, \mathcal{A},\mathcal{B} \in \Phi.$	$P \models \mathcal{A} \mid \mathcal{B}$	$\triangleq$	$\exists P', P'' \in \Pi.\ P \equiv P' \mid P'' \wedge P' \models \mathcal{A} \wedge P'' \models \mathcal{B}$
$\forall P \in \Pi, \mathcal{A},\mathcal{B} \in \Phi.$	$P \models \mathcal{A} \triangleright \mathcal{B}$	$\triangleq$	$\forall P' \in \Pi.\ P' \models \mathcal{A} \Rightarrow P \mid P' \models \mathcal{B}$
$\forall P \in \Pi, n \in \Lambda, \mathcal{A} \in \Phi.$	$P \models n[\mathcal{A}]$	$\triangleq$	$\exists P' \in \Pi.\ P \equiv n[P'] \wedge P' \models \mathcal{A}$
$\forall P \in \Pi, \mathcal{A} \in \Phi.$	$P \models \mathcal{A}@n$	$\triangleq$	$n[P] \models \mathcal{A}$
$\forall P \in \Pi, n \in \Lambda, \mathcal{A} \in \Phi.$	$P \models n \circledR \mathcal{A}$	$\triangleq$	$\exists P' \in \Pi.\ P \equiv (\nu n)P' \wedge P' \models \mathcal{A}$
$\forall P \in \Pi, \mathcal{A} \in \Phi.$	$P \models \mathcal{A} \oslash n$	$\triangleq$	$(\nu n)P \models \mathcal{A}$
$\forall P \in \Pi, \mathcal{A} \in \Phi.$	$P \models \Diamond\mathcal{A}$	$\triangleq$	$\exists P' \in \Pi.\ P \rightarrow^* P' \wedge P' \models \mathcal{A}$
$\forall P \in \Pi, \mathcal{A} \in \Phi.$	$P \models \diamondsuit\mathcal{A}$	$\triangleq$	$\exists P' \in \Pi.\ P \downarrow^* P' \wedge P' \models \mathcal{A}$
$\forall P \in \Pi, x \in \vartheta, \mathcal{A} \in \Phi.$	$P \models \forall x.\mathcal{A}$	$\triangleq$	$\forall m \in \Lambda.\ P \models \mathcal{A}\{x \leftarrow m\}$

The first three connectives gives classical propositional logic. Any process satisfies the formula **T**. A process satisfies the formula $\neg\mathcal{A}$ if it does not satisfy the formula $\mathcal{A}$. A process satisfies the formula $\mathcal{A} \vee \mathcal{B}$ if it satisfies either the formula $\mathcal{A}$ or the formula $\mathcal{B}$.

The next group form the core of the process logic. A process P satisfies the formula $\mathbf{0}$ if $P \equiv \mathbf{0}$. A process P satisfies the formula $\mathcal{A}' \mid \mathcal{A}''$ if there exist processes P' and P'' such that P has the shape $P' \mid P''$ with P' satisfying $\mathcal{A}'$ and P'' satisfying $\mathcal{A}''$. A process P satisfies the formula $n[\mathcal{A}]$ if there exists a process P' such that P has the shape $n[P']$ with P' satisfying $\mathcal{A}$. The connectives @ and $\triangleright$, can be used to express context/system specifications; they were inspired by the wish to express security properties. A reading of $P \models \mathcal{A}@n$ is that P (together with its context) manages to satisfy $\mathcal{A}$ even when placed into a location called n. A reading of $P \models \mathcal{A} \triangleright \mathcal{B}$ is that P (together with its context) manages to satisfy $\mathcal{B}$ under

any possible attack by an opponent that is bound to satisfy $\mathcal{A}$. We will see that these two connectives arise as natural adjuncts to the location and composition connectives.

The connective ® is used to *reveal* a restricted name. A process P satisfies the formula $n®\mathcal{A}$ if it is possible to pull a restricted name of P to the top and call it n (this is not possible if n is already free in P), and then strip off the restriction to leave a residual that satisfies $\mathcal{A}$. Examples: $(\nu n)n[] \vDash n®\mathbf{T}$, $\mathbf{0} \vDash n®\mathbf{T}$, $\neg\, n[] \vDash n®\mathbf{T}$, $(\nu m)m[] \vDash n®n[]$. The connective $\oslash$ is used to *hide* (restrict) a name: a process P satisfies the formula $\mathcal{A}\oslash n$ if $(\nu n)P$ satisfies $\mathcal{A}$; this connective arises as a natural adjunct to ®. Revelation and hiding are discussed further in Section 5.

A process P satisfies the formula $\Diamond\mathcal{A}$ if $\mathcal{A}$ holds in the future for some residual P' of P, where "residual" is defined by $P \rightarrow^* P'$. A process P satisfies the formula $\diamond\mathcal{A}$ if $\mathcal{A}$ holds at some sublocation P' within P, where "sublocation" is defined by $P \downarrow^* P'$.

Universal quantification ranges just over names. A process P satisfies the formula $\forall x.\mathcal{A}$ if for all names m we have that P satisfies $\mathcal{A}\{x \leftarrow m\}$.

Remark: Given our policy of identifying formulas up to the renaming of bound variables, we need to check that satisfaction is well defined with respect to the equation $\forall x.\mathcal{A} = \forall y.\mathcal{A}\{x \leftarrow y\}$, where $y \notin fv(\mathcal{A})$. We need to show for all processes P, formulas $\mathcal{A}$, and variables x and y such that $y \notin fv(\mathcal{A})$ that $P \vDash \forall x.\mathcal{A}$ if and only if $P \vDash \forall y.\mathcal{A}\{x \leftarrow y\}$. By definition, $P \vDash \forall y.\mathcal{A}\{x \leftarrow y\}$ if and only if $\forall m \in \Lambda.\ P \vDash \mathcal{A}\{x \leftarrow y\}\{y \leftarrow m\}$. Since $y \notin fv(\mathcal{A})$, we have $\mathcal{A}\{x \leftarrow y\}\{y \leftarrow m\} = \mathcal{A}\{x \leftarrow m\}$. Therefore, $P \vDash \forall y.\mathcal{A}\{x \leftarrow y\}$ if and only if $\forall m \in \Lambda.\ P \vDash \mathcal{A}\{x \leftarrow m\}$ This is the definition of satisfaction for $\forall x.\mathcal{A}$. So it follows that $y \notin fv(\mathcal{A})$ implies that $P \vDash \forall x.\mathcal{A}$ if and only if $P \vDash \forall y.\mathcal{A}\{x \leftarrow y\}$. □

The following table lists some derived connectives, and it illustrates properties that can be expressed in the logic. The informal meanings can be understood better by expanding out the definitions from the table above. See also Section 3.3 for examples.

Derived Connectives

$\mathbf{F}$	$\triangleq \neg\mathbf{T}$	false
$\mathcal{A} \wedge \mathcal{B}$	$\triangleq \neg(\neg\mathcal{A} \vee \neg\mathcal{B})$	conjunction
$\mathcal{A} \Rightarrow \mathcal{B}$	$\triangleq \neg\mathcal{A} \vee \mathcal{B}$	implication
$\mathcal{A} \Leftrightarrow \mathcal{B}$	$\triangleq (\mathcal{A} \Rightarrow \mathcal{B}) \wedge (\mathcal{B} \Rightarrow \mathcal{A})$	logical equivalence
$\mathcal{A} \parallel \mathcal{B}$	$\triangleq \neg(\neg\mathcal{A} \mid \neg\mathcal{B})$	decomposition
$\mathcal{A}^{\forall}$	$\triangleq \mathcal{A} \parallel \mathbf{F}$	every component satisfies $\mathcal{A}$
$\mathcal{A}^{\exists}$	$\triangleq \mathcal{A} \mid \mathbf{T}$ $\quad (\Leftrightarrow \neg(\neg\mathcal{A}^{\forall}))$	some component satisfies $\mathcal{A}$
$\exists x.\mathcal{A}$	$\triangleq \neg\forall x.\neg\mathcal{A}$	existential quantification over names

$\Box\mathcal{A}$	$\triangleq \neg\Diamond\neg\mathcal{A}$	everytime modality
$\boxtimes\mathcal{A}$	$\triangleq \neg\diamond\neg\mathcal{A}$	everywhere modality
$\mathcal{A} \propto \mathcal{B}$	$\triangleq \neg(\mathcal{B} \triangleright \neg\mathcal{A})$	fusion
$\mathcal{A} \mid\Rightarrow \mathcal{B}$	$\triangleq \neg(\mathcal{A} \mid \neg\mathcal{B})$	fusion adjunct
$n[\Rightarrow\mathcal{A}]$	$\triangleq \neg n[\neg\mathcal{A}]$	if there is an n, its contents satisfy $\mathcal{A}$
$\mathcal{A}^{\mathbf{F}}$	$\triangleq \mathcal{A}\triangleright\mathbf{F}$	$\mathcal{A}$ is unsatisfiable
$\mathcal{A}\urcorner$	$\triangleq \neg\mathcal{A}$	$\mathcal{A}$ is false (it is not the full set of processes)
$\copyright\eta$	$\triangleq \neg\eta\circledR\mathbf{T}$	contains η

Some syntactic conventions: Parentheses are used for explicit precedence. '$\triangleright$' binds more strongly than '$\mid$'; they both bind more strongly than the standard logical connectives, which have standard precedences. Quantifiers and modalities extend to the right as far as possible. Postfix, '$^{\mathbf{F}}$' binds more strongly than prefix '$\neg$'; moreover, in conjunction with $\mathcal{A}^{\mathbf{F}}$ we use the notation $\mathcal{A}\urcorner$ for $\neg\mathcal{A}$.

Some Expanded Definitions

$\forall P{:}\Pi.$	$\neg P \models \mathbf{F}$	
$\forall P{:}\Pi, \mathcal{A},\mathcal{B}{:}\Phi.$	$P \models \mathcal{A}\wedge\mathcal{B}$	iff $P \models \mathcal{A} \wedge P \models \mathcal{B}$
$\forall P{:}\Pi, \mathcal{A},\mathcal{B}{:}\Phi.$	$P \models \mathcal{A}\Rightarrow\mathcal{B}$	iff $P \models \mathcal{A} \Rightarrow P \models \mathcal{B}$
$\forall P{:}\Pi, \mathcal{A},\mathcal{B}{:}\Phi.$	$P \models \mathcal{A}\Leftrightarrow\mathcal{B}$	iff $P \models \mathcal{A} \Leftrightarrow P \models \mathcal{B}$
$\forall P{:}\Pi, \mathcal{A},\mathcal{B}{:}\Phi.$	$P \models \mathcal{A} \parallel \mathcal{B}$	iff $\forall P',P''{:}\Pi.\ P \equiv P'\mid P'' \Rightarrow P' \models \mathcal{A} \vee P'' \models \mathcal{B}$
$\forall P{:}\Pi, \mathcal{A}{:}\Phi.$	$P \models \mathcal{A}^{\forall}$	iff $\forall P',P''{:}\Pi.\ P \equiv P'\mid P'' \Rightarrow P' \models \mathcal{A}$
$\forall P{:}\Pi, \mathcal{A}{:}\Phi.$	$P \models \mathcal{A}^{\exists}$	iff $\exists P',P''{:}\Pi.\ P \equiv P'\mid P'' \wedge P' \models \mathcal{A}$
$\forall P{:}\Pi, x{:}\vartheta, \mathcal{A}{:}\Phi.$	$P \models \exists x.\mathcal{A}$	iff $\exists m{:}\Lambda.\ P \models \mathcal{A}\{x\leftarrow m\}$
$\forall P{:}\Pi, \mathcal{A}{:}\Phi.$	$P \models \Box\mathcal{A}$	iff $\forall P'{:}\Pi.\ P \rightarrow^{*} P' \Rightarrow P' \models \mathcal{A}$
$\forall P{:}\Pi, \mathcal{A}{:}\Phi.$	$P \models \boxtimes\mathcal{A}$	iff $\forall P'{:}\Pi.\ P\downarrow^{*}P' \Rightarrow P' \models \mathcal{A}$
$\forall P{:}\Pi, \mathcal{A},\mathcal{B}{:}\Phi.$	$P \models \mathbf{T}\triangleright(\mathcal{A}\Rightarrow\mathcal{B})$	iff $\forall P'{:}\Pi.\ P'\mid P \models \mathcal{A} \Rightarrow P'\mid P \models \mathcal{B}$ (*cf.* $P \models \mathcal{A}\triangleright\mathcal{B}$)
$\forall P{:}\Pi, \mathcal{A},\mathcal{B}{:}\Phi.$	$P \models \mathcal{A}\propto\mathcal{B}$	iff $\exists P'{:}\Pi.\ P' \models \mathcal{B} \wedge P\mid P' \models \mathcal{A}$
$\forall P{:}\Pi, \mathcal{A},\mathcal{B}{:}\Phi.$	$P \models \mathcal{A} \mid\Rightarrow \mathcal{B}$	iff $\forall P',P''{:}\Pi.\ P \equiv P'\mid P'' \Rightarrow (P' \models \mathcal{A} \Rightarrow P'' \models \mathcal{B})$
$\forall P{:}\Pi, \mathcal{A}{:}\Phi.$	$P \models n[\Rightarrow\mathcal{A}]$	iff $\forall P'{:}\Pi.\ P \equiv n[P'] \Rightarrow P' \models \mathcal{A}$
$\forall P{:}\Pi, \mathcal{A}{:}\Phi.$	$P \models \mathcal{A}^{\mathbf{F}}$	iff $\forall P'{:}\Pi.\ \neg P' \models \mathcal{A}$
$\forall P{:}\Pi, \mathcal{A}{:}\Phi.$	$P \models \mathcal{A}^{\mathbf{F}\neg}$	iff $\exists P'{:}\Pi.\ P' \models \mathcal{A}$
$\forall P{:}\Pi, \mathcal{A}{:}\Phi.$	$P \models \mathcal{A}\urcorner^{\mathbf{F}}$	iff $\forall P'{:}\Pi.\ P' \models \mathcal{A}$
$\forall P{:}\Pi.$	$P \models \copyright n$	iff $\neg\exists P'{:}\Pi.\ P \equiv (\nu n)P'$ iff $n\in fn(P)$

The derived logical connectives can be read as follows:

- No process satisfies the formula **F**.
- A process satisfies the formula $\mathcal{A} \wedge \mathcal{B}$ if it satisfies both the formulas $\mathcal{A}$ and $\mathcal{B}$.
- A process satisfies the formula $\mathcal{A} \Rightarrow \mathcal{B}$ if either it does not satisfy the formula $\mathcal{A}$ or it satisfies the formula $\mathcal{B}$.

- A process satisfies the formula $\mathcal{A} \Leftrightarrow \mathcal{B}$ if it satisfies neither or both the formulas $\mathcal{A}$ and $\mathcal{B}$.
- A process P satisfies the formula $\mathcal{A}' \parallel \mathcal{A}''$ if for every decomposition of P into processes P' and P'' such that $P \equiv P' \mid P''$, either P' satisfies $\mathcal{A}'$ or P'' satisfies $\mathcal{A}''$. That is, for every split, either one part satisfies $\mathcal{A}'$ or the other satisfies $\mathcal{A}''$.
- A process P satisfies the formula $\mathcal{A}^{\forall}$ if every parallel component P' of P (such that $P \equiv P' \mid P''$, including $P' = \mathbf{0}$) satisfies the formula $\mathcal{A}$. For example, "each n location contains just an m location" can be written: $(n[\mathbf{T}] \Rightarrow n[m[\mathbf{T}]])^{\forall}$
- A process P satisfies the formula $\mathcal{A}^{\exists}$ if there is a parallel component P' of P (such that $P \equiv P' \mid P''$) that satisfies the formula $\mathcal{A}$. For example, "there is an n containing an m" can be written: $n[m[\mathbf{T}]^{\exists}]^{\exists}$.
- A process P satisfies the formula $\exists x.\mathcal{A}$ if there is a name m such that P satisfies $\mathcal{A}\{x \leftarrow m\}$. For example, "there is a location contained into a location with the same name" can be written: $\exists x.x[x[\mathbf{T}]]$
- A process P satisfies the formula $\Box\mathcal{A}$ if $\mathcal{A}$ holds in the future for every residual P' of P, where "residual" is defined by $P \rightarrow^{*} P'$. For example, "there is always a location n" can be written: $\Box n[\mathbf{T}]$.
- A process P satisfies the formula $\boxtimes\mathcal{A}$ if $\mathcal{A}$ holds at every sublocation P' within P, where "sublocation" is defined by $P \downarrow^{*} P'$. For example, "there is no location n can be written: $\boxtimes\neg(n[\mathbf{T}]^{\exists})$.
- If process P satisfies the formula $\mathcal{A} \triangleright \mathcal{B}$, it means that in every context that satisfies $\mathcal{A}$, the combination of P and the context satisfies $\mathcal{B}$. Instead, for example, if process P satisfies the formula $\mathbf{T} \triangleright (\mathcal{A} \Rightarrow \mathcal{B})$, it means that in every context, if the combination satisfies $\mathcal{A}$ then the combination satisfies $\mathcal{B}$.
- A process P satisfies the formula $\mathcal{A} \propto \mathcal{B}$ if there is a process P' satisfying $\mathcal{B}$ that can be composed with P to satisfy $\mathcal{A}$.
- A process P satisfies the formula $\mathcal{A} \mid\Rightarrow \mathcal{B}$ if for every decomposition of P, if one part satisfies $\mathcal{A}$ then the other part satisfies $\mathcal{B}$.
- A process P satisfies $n[\Rightarrow\mathcal{A}]$ iff in case P is a location called n, its contents satisfy $\mathcal{A}$.
- A process P satisfies the formula $\mathcal{A}^{\mathbf{F}}$ if no process satisfies $\mathcal{A}$. A process P satisfies the formula $\mathcal{A}^{\mathbf{F}\neg}$ if there is a process that satisfies $\mathcal{A}$. A process P satisfies the formula $\mathcal{A}^{\neg\mathbf{F}}$ if every process satisfies $\mathcal{A}$.
- A process P satisfies the formula $\copyright n$ if $n \in fn(P)$. Examples: $(x).n[] \models \copyright n$, $0 \models \neg\copyright n$.

Fundamental Lemmas

The following lemmas are crucial in what follows.

3-1 Lemma (Satisfaction is up to ≡)

$(P \models \mathcal{A} \wedge P \equiv P') \Rightarrow P' \models \mathcal{A}$

Proof

A simple induction on the structure of $\mathcal{A}$.

□

3-2 Lemma (Fresh renaming preserves ≡)

Consider any process P and names m, m', with $m' \notin fn(P)$. For all P', if $P \equiv P'$ then $m' \notin fn(P')$ and $P\{m \leftarrow m'\} \equiv P'\{m \leftarrow m'\}$. Moreover, for all Q, if $P\{m \leftarrow m'\} \equiv Q$ then there is a P' with $P \equiv P'$, $m' \notin fn(P)$ and $Q = P'\{m \leftarrow m'\}$.

□

3-3 Lemma (Fresh renaming preserves ⊨) (Proof in the Appendix)

For all closed formulas $\mathcal{A}$, processes P, and names m, m',
if $m' \notin fn(P) \cup fn(\mathcal{A})$ then $P \models \mathcal{A} \Leftrightarrow P\{m \leftarrow m'\} \models \mathcal{A}\{m \leftarrow m'\}$.

□

A proof of a negative formula

The proof of even a very simple negative formula requires techniques for analyzing the derivation of structural congruences. For example, consider proving the following assertion, where $m \neq n$:

$$m[] \mid n[] \models \neg\, \exists x.\, x[\mathbf{T}] \mid x[\mathbf{T}]$$

For a contradiction, suppose that $m[] \mid n[] \models \exists x.\, x[\mathbf{T}] \mid x[\mathbf{T}]$. By definition, this means there is a P such that $m[] \mid n[] \equiv P$ and there is a q with $P \models q[\mathbf{T}] \mid q[\mathbf{T}]$. This implies that there are processes P' and P'' such that $m[] \mid n[] \equiv P' \mid P''$ with $P' \models q[\mathbf{T}]$ and $P'' \models q[\mathbf{T}]$. In turn, $P' \models q[\mathbf{T}]$ implies there is Q' such that $P' \equiv q[Q']$. Similarly, $P'' \models q[\mathbf{T}]$ implies there is Q'' such that $P'' \equiv q[Q'']$. In summary, we have:

$$m[] \mid n[] \equiv q[Q'] \mid q[Q'']$$

According to the Lemmas 2-1, there are two ways in which this equation can have been derived. In either case, it follows that $m = q$ and $n = q$, and therefore $m = n$. This yields the desired contradiction, as we are assuming that $m \neq n$.

3.3 Expressiveness

In this section we provide simple examples of properties that can be stated within the logic, and that can be verified against specific processes.

Mobility

The process:

$$P \triangleq a[m[out\ a.\ in\ b.\ \langle c \rangle]] \mid b[open\ m.\ (n).\ n[]]$$

represents a message m with payload c being routed from location a to location b. The message is opened when it reaches b; the payload c is read into the bound name n, and an empty ambient $c[]$ is then produced inside of b. Hence this process reduces to:

$$Q \triangleq a[] \mid b[c[]]$$

Using the satisfaction semantics of Section 3.2, we can check that various assertions hold of P and its reduced form Q. For example:

$P \models a[\mathbf{T}] \mid b[\mathbf{T}] \mid \mathbf{T}$	P includes locations a and b
$P \models a[m[\mathbf{T}]] \mid \mathbf{T}$	there is a (message) m in a
$P \models \Box\Diamond(b[m[\mathbf{T}]] \mid \mathbf{T})$	a (message) m will be found in b
$P \models \Box\Diamond\Diamond c[]$	an empty location c will be produced

A typical protocol specification consists of a conjunction of two assertions, one about the initial state of the system and one about its possible or necessary final state. For example, here we could state $(a[m[\mathbf{T}]] \mid \mathbf{T}) \wedge \Box\Diamond(b[m[\mathbf{T}]] \mid \mathbf{T})$ meaning that there is now an m inside a, and that eventually there will be an m inside b. It is not possible to say directly that these two m's are the "same" ambient, but one can enrich the specification to say, for example, that initially there is no m inside b and finally there is no m inside a.

Revelation

The revelation operator $\eta \circledR \mathcal{A}$ can be used, in particular, to express the occurrence of free names in processes. We start by defining a derived formula $\copyright\eta$:

$$\copyright\eta \triangleq \neg\eta \circledR \mathbf{T} \qquad \text{meaning: contains the name } \eta \text{ free}$$

We can then state properties such as the following:

closed	$\triangleq \neg\exists x.\copyright x$	has no free names
separate	$\triangleq \neg\exists x.\copyright x \mid \copyright x$	has no shared free names
atmostfree η	$\triangleq$ *closed* $\oslash \eta$	has at most η as a free name

For clarity, we expand the definitions:

$$\begin{array}{llll}
\forall P{\in}\Pi. & P \models \copyright n & \text{iff } \neg\exists P'{\in}\Pi.\ P \equiv (\nu n)P' & \text{iff } n{\in}fn(P) \\
\forall P{\in}\Pi. & P \models \textit{closed} & \text{iff } \forall n{\in}\Lambda.\ \exists P'{\in}\Pi.\ P \equiv (\nu n)P' & \text{iff } fn(P) = \emptyset \\
\forall P{\in}\Pi. & P \models \textit{separate} & \text{iff } \neg\exists n{\in}\Lambda.\ \exists P',P''{\in}\Pi.\ P \equiv P' \mid P'' \wedge n{\in}fn(P') \wedge n{\in}fn(P'') & \\
\forall P{\in}\Pi. & P \models \textit{atmostfree } n & \text{iff } \forall m{\in}\Lambda.\ \exists P'{\in}\Pi.\ (\nu n)P \equiv (\nu m)P' & \text{iff } fn(P) \subseteq \{n\}
\end{array}$$

Examples:

$n[] \models \copyright n$	because $\neg\exists P'{\in}\Pi.\ n[] \equiv (\nu n)P'$
$(\nu m)m[] \models$ *closed*	because $\forall n{\in}\Lambda.\ (\nu m)m[] \equiv (\nu n)(\nu m)m[]$
$n[] \mid m[] \mid (\nu p)(p[] \mid p[]) \models$ *separate*	
$n[] \models$ *atmostfree n*	because $(\nu n)n[] \models$ *closed*

Name Equality

Name equality can be expressed within the logic (Section 4.7), by setting:

$$\eta = \mu \;\triangleq\; \eta[\mathbf{T}]@\mu$$

We can check that $P \models n[\mathbf{T}]@m$ iff $m[P] \models n[\mathbf{T}]$ iff $m = n$, and this is independent of the choice of P. As an example, the following formula means "any two top-level ambients have different names", which can be interpreted as a no-spoofing security property:

$$\textit{top-distinct} \;\triangleq\; \forall x.\, \forall y.\, x[\mathbf{T}] \mid y[\mathbf{T}] \mid \mathbf{T} \Rightarrow \neg\, x = y$$

and we can verify that $n[m[]] \mid m[] \models \textit{top-distinct}$ (while $n[m[]] \mid m[] \not\models \textit{separate}$).

Hiding

Let us now consider the process:

$$P \;\triangleq\; (\nu p)p[n[]]$$

where the ambient $n[]$ is the only ambient contained inside a hidden location. The name of such a hidden location cannot be mentioned, because it is bound by restriction. Nonetheless, suppose we want to describe formally such a property of P; we can write:

$$P \models \exists x.\, x{\neq}n \wedge \neg{\copyright}x \wedge x{\circledR}x[n[\mathbf{T}]]$$

meaning that there is a name x (which in this particular example can be taken to be any $p{\neq}n$), such that $x{\neq}n$ (that is, the hidden name does not clash with the public name n), and $\neg{\copyright}x$ (the hidden name does not accidentally appear free in P), and $x{\circledR}x[n[\mathbf{T}]]$ (P consists of a restriction, which once opened by using x as the hidden name, reveals the structure $x[n[\mathbf{T}]]$).

The entire formula is read as a whole as "there is a hidden name x such that $x[n[\mathbf{T}]]$" and is better motivated in Sections 6 and 7 in terms of a hidden-name quantifier. For an example of satisfaction for hidden name quantifiers, see Section 7.3.

4 Validity

In this section, we study valid formulas, valid sequents, and valid logical inference rules. All these are based on the satisfaction relation given in the previous section. Once the definition of satisfaction is fixed, we are basically committed to whatever logic comes out of it. Therefore, it is important to stress that the satisfaction relation appears very natural to us. In particular, the definitions of $\mathbf{0}$, $n[\mathcal{A}]$, and $\mathcal{A} \mid \mathcal{B}$ seem inevitable, once we accept that formulas should be able to talk about the tree structure of locations, and that they should not distinguish processes that are surely indistinguishable (up to $\equiv$). The connectives $\mathcal{A}@n$ and $\mathcal{A}{\triangleright}\mathcal{B}$ have natural security motivations. The modalities $\Diamond\mathcal{A}$ and $\Cross\mathcal{A}$ talk about process evolution and structure in an undetermined way, which is good for mobility specifications. The rest is classical predicate logic, with the ability to quantify over location names. The connectives ® and ⦸ are perhaps the least natural; they are discussed and motivated in Section 5.

Through the satisfaction relation, our logic is based on solid computational intuitions. We should now approach the task of discovering the rules of the logic without preconceptions. As we shall see, what we get has familiar as well as novel aspects.

We adopt a non-standard formulation of sequents, where each sequent has exactly one assumption and one conclusion: $\mathcal{A} \vdash \mathcal{B}$. Our intention in doing so is to avoid pre-judging the interpretation of the structural operator "," in standard sequents. In our logic, by taking $\wedge$ on the left and $\vee$ on the right of $\vdash$ as structural operators (i.e., as ","), all the standard rules of sequent and natural deduction systems with multiple premises/conclusions can be derived. Instead, by taking | on the left of $\vdash$ as a structural operator, all the rules of intuitionistic linear logic can be derived. Finally, by taking nestings of $\wedge$ and | on the left of $\vdash$ as structural "bunches", we obtain a bunched logic [31]. We discuss this further in Section 8. This form of sequents contrasts with the one adopted in [5], in related work.

In the sequel, we organize our results into Validity Propositions for inference rules that are validated in the model, and into Logical Corollaries (mostly found in the Appendix) that are derived purely logically from the inference rules. Both primitive and derived rules are given individual names; they are referred to by those names, not by Proposition or Corollary number. Many rules have names of the form (- $\vdash$); these are *entailment* rules that generally relate entailment of logical connectives to entailment of their arguments. Theses are akin to congruence rules over the syntax of logical connectives: by composing these rules one can substitute equals for equals into formulas.

Valid Formulas, Sequents, and Rules

A closed formula is valid when it is satisfied by all processes. A general formula is valid when it is valid under any closed instantiation of its free variables with names.

More precisely, if $fv(\mathcal{A})=\{x_1, ..., x_k\}$ are the free variables of $\mathcal{A}$ and $\varphi \in \vartheta \rightarrow \Lambda$ is a substitution of names for variables such that $dom(\varphi) \supseteq fv(\mathcal{A})$, then we write $\mathcal{A}_\varphi$ for $\mathcal{A}\{x_1 \leftarrow \varphi(x_1), ..., x_k \leftarrow \varphi(x_k)\}$, and we define:

Valid Formulas

$$\textbf{\textit{vld}}(\mathcal{A})_\varphi \triangleq \forall P \in \Pi.\ P \vDash \mathcal{A}_\varphi \quad \text{for } \varphi \in \vartheta \rightarrow \Lambda \text{ with } dom(\varphi) \supseteq fv(\mathcal{A})$$
$$\textbf{\textit{vld}}(\mathcal{A}) \triangleq \forall \varphi \in fv(\mathcal{A}) \rightarrow \Lambda.\ \textbf{\textit{vld}}(\mathcal{A})_\varphi$$

We use validity for interpreting logical inference rules, as described in the following tables. We use a linearized notation for inference rules, where the usual horizontal bar separating antecedents from consequents is written '$\upharpoonright$' in-line, and ';' is used to separate antecedents.

Sequents are interpreted as follows. A simple sequent $\mathcal{A} \vdash \mathcal{B}$ is interpreted as the validity of the formula $\mathcal{A} \Rightarrow \mathcal{B}$. Sequents with conditions about disjointness of variables, or disjointness of variables from names, are reduced to simple sequents, as described below. Note that, as discussed in Section 4.7, equality of names $\eta=\mu$ is definable in the logic as $\eta[\mathbf{T}]@\mu$.

Sequents

$$\mathcal{A} \vdash \mathcal{B} \triangleq \textbf{\textit{vld}}(\mathcal{A} \Rightarrow \mathcal{B})$$
$$\mathcal{A} \vdash \mathcal{B}\ (\eta_1 \neq \mu_1, ..., \eta_n \neq \mu_n) \triangleq (\eta_1 \neq \mu_1 \wedge ... \wedge \eta_n \neq \mu_n \wedge \mathcal{A}) \vdash \mathcal{B}$$
$$\mathcal{A} \dashv\vdash \mathcal{B}\ (\Xi) \triangleq (\mathcal{A} \vdash \mathcal{B}\ (\Xi)) \wedge (\mathcal{B} \vdash \mathcal{A}\ (\Xi)) \quad \text{where } \Xi = \eta_1 \neq \mu_1, ..., \eta_n \neq \mu_n$$

For example: $\mathcal{A} \vdash \mathcal{B}$ means $\forall \varphi \in fv(\mathcal{A} \Rightarrow \mathcal{B}) \rightarrow \Lambda.\ \forall P \in \Pi.\ P \models \mathcal{A}_\varphi \Rightarrow P \models \mathcal{B}_\varphi$. To be precise, a given sequent cannot contain instances of the metavariables η and μ, but it may contain either a name or a variable where indicated by η and μ.

Logical rules are interpreted as follows, where $\mathcal{S}$ are sequents (any of the three forms above, including sequents with side conditions and double sequents):

Rules

$\mathcal{S}_1; ...; \mathcal{S}_n \nmid \mathcal{S}_0 \quad \triangleq \quad (\mathcal{S}_1 \wedge ... \wedge \mathcal{S}_n) \Rightarrow \mathcal{S}_0$
$\mathcal{S}_1 \nmid\!\mid \mathcal{S}_2 \quad \triangleq \quad \mathcal{S}_1 \nmid \mathcal{S}_2 \ \wedge \ \mathcal{S}_2 \nmid \mathcal{S}_1$

The definition of validity for formulas with free variables allows us to handle quantification over names. We obtain the validity of the following standard rules for the universal quantifier, and for the definable existential quantifier:

Quantification

($\forall$ L)	$\mathcal{A}\{x \leftarrow \eta\} \vdash \mathcal{B} \ \nmid \ \forall x.\mathcal{A} \vdash \mathcal{B}$	where η is a name or a variable
($\forall$ R)	$\mathcal{A} \vdash \mathcal{B} \ \nmid \ \mathcal{A} \vdash \forall x.\mathcal{B}$	where $x \notin fv(\mathcal{A})$
($\exists$ L)	$\mathcal{A} \vdash \mathcal{B} \ \nmid \ \exists x.\mathcal{A} \vdash \mathcal{B}$	where $x \notin fv(\mathcal{B})$
($\exists$ R)	$\mathcal{A} \vdash \mathcal{B}\{x \leftarrow \eta\} \ \nmid \ \mathcal{A} \vdash \exists x.\mathcal{B}$	where η is a name or a variable

***Remark:* ($\forall$ R).** The distinction between variables and names in formulas, and the use of variables (as opposed to names) in quantification is crucial for ($\forall$ R). The version of ($\forall$ R) with names instead of variables:

$$\mathcal{A} \vdash \mathcal{B} \ \nmid \ \mathcal{A} \vdash \forall n.\mathcal{B} \qquad \text{where } n \notin fn(\mathcal{A}),$$

is not sound. Consider the valid sequent $m[\mathbf{T}] \vdash \neg n[\mathbf{T}]$. If quantification binders were names, then the rule ($\forall$ R) could be used to produce $m[\mathbf{T}] \vdash \forall n.\neg n[\mathbf{T}]$, which is not valid. Since quantification binders are variables, one can only deduce $m[\mathbf{T}] \vdash \forall x.\neg n[\mathbf{T}]$. □

***Remark:* ($\forall$ L).** The use of substitutions that admit variables, in addition to names, in ($\forall$ L), is crucial. Otherwise, if ($\forall$ L) is formulated as $\mathcal{A}\{x \leftarrow m\} \vdash \mathcal{B} \ \nmid \ \forall x.\mathcal{A} \vdash \mathcal{B}$, there does not seem to be any way to derive, for example:

$$\mathcal{A} \vdash \mathcal{B} \ \nmid \ \forall x.\mathcal{A} \vdash \forall x.\mathcal{B}$$

which is obtained by starting from $\mathcal{A}\{x \leftarrow x\} \vdash \mathcal{B}$ and applying ($\forall$ L) and then ($\forall$ R). □

Remark: Scope of variables in rules. Free variables are independently quantified in each sequent of a rule, and not across the whole rule. (This is necessary for the soundness of ($\forall$ R).) For example, the rule $\mathcal{A}_1 \vdash \mathcal{B}_1 \nmid \mathcal{A}_0 \vdash \mathcal{B}_0$ means: $(\forall \varphi \in fv(\mathcal{A}_1 \Rightarrow \mathcal{B}_1) \rightarrow \Lambda.\ \boldsymbol{vld}(\mathcal{A}_1 \Rightarrow \mathcal{B}_1)_\varphi) \Rightarrow (\forall \varphi \in fv(\mathcal{A}_0 \Rightarrow \mathcal{B}_0) \rightarrow \Lambda.\ \boldsymbol{vld}(\mathcal{A}_0 \Rightarrow \mathcal{B}_0)_\varphi)$. Therefore, there is no actual relationship between identical variables occurring on the left and on the right of $\nmid$. □

Remark: Name and Variable form for Axioms. When an axiom (a rule without antecedents) is meant to hold for both variables and names, we present it with variables, and we rely on an instantiation principle (Proposition 4-1) to derive the form with names. Conversely, an axiom can be systematically lifted from name form to variable form by a technique from [13] (section 4.2.8), which introduces side conditions about disjointness of variables. However, that technique often introduces unnecessary side conditions. □

Remark: Name and Variable form for Rules. When a full rule (with antecedents) is meant to hold for both variables and names, the situation gets subtle, particularly if related variables or names occur throughout the rule. Consider one of the rules we use later, expressed in variable and name form; both forms are valid:

(1) $x \circledR \mathcal{A} \vdash \mathcal{B} \;\}\; \mathcal{A} \vdash \mathcal{B} \oslash x$
(2) $n \circledR \mathcal{A} \vdash \mathcal{B} \;\}\; \mathcal{A} \vdash \mathcal{B} \oslash n$

In rule (1) the two x's are under separate φ quantifications, so this really means the same as $x \circledR \mathcal{A} \vdash \mathcal{B} \;\}\; \mathcal{A}\{x \leftarrow y\} \vdash \mathcal{B}\{x \leftarrow y\} \oslash y$ for a fresh y. Rule (1) by itself is not enough, because (2) is not derivable from (1) by the instantiation principle, and we need (2) for deductions that involve names. On the other hand, rule (2) by itself is not enough for deductions that involve variables. For example, from the axiom $\;\}\; x \circledR x \circledR \mathcal{A} \vdash x \circledR \mathcal{A}$ we may want to derive $x \circledR \mathcal{A} \vdash (x \circledR \mathcal{A}) \oslash x$ by (1), and then $\forall x.\, x \circledR \mathcal{A} \vdash \forall x.\, (x \circledR \mathcal{A}) \oslash x$. This conclusion cannot be derived if we do not start with variables in the first place, because otherwise we could never use ($\forall$ R). Therefore, we need both (1) and (2). A compact way to write these two rules is:

(3) $\eta \circledR \mathcal{A} \vdash \mathcal{B} \;\}\; \mathcal{A} \vdash \mathcal{B} \oslash \eta$

However, we should remember that this is just an abbreviation for (1) and (2). If we had two metavariables η and μ in a rule, this would represent four rules (although we have no need for this at the moment). □

Remark: Schematic Side Conditions. The disjointness side conditions, e.g.:

$\mathcal{A} \vdash \mathcal{B}\ (x \# y) \quad \triangleq \quad (x \# y \wedge \mathcal{A}) \vdash \mathcal{B}$

are interpreted within the logic, and always impose restrictions on the *free* variables or names of a sequent. They really express restrictions on the allowable instantiations of free variables to names, and only indirectly on disjointness of variables. A different kind of side condition is *schematic*; it can impose restrictions on bound variables, and it arises uniquely from the side condition to the rule ($\forall$ R): $x \notin fv(\mathcal{A})$, which talks about an actual variable, and not about its allowable instantiations. These schematic side conditions restrict, metatheoretically, the legal instances of a rule, and we always prefix them by "where". In some cases a schematic side condition can look very similar to a disjointness side condition, and this can be a bit confusing. For example, consider the following derivable sequent (an instance of a derivable rule we later call ($\circledR\ \forall$)), where y is free and x is bound:

$\;\}\; y \circledR \forall x.x[] \vdash \forall x.y \circledR x[] \quad$ where $x \neq y$

To explain what this side condition means, we examine the derivation, beginning with:

$\}_{(\mathrm{Id})} \;\; x[] \vdash x[] \;=\; x[]\{x \leftarrow x\} \vdash x[]$
$\}_{(\forall\,\mathrm{L})} \;\; \forall x.x[] \vdash x[]$

At this point we could legally apply ($\circledR \vdash$) using x, in the following way:

$\}_{(\circledR \vdash)} \;\; x \circledR \forall x.x[] \vdash x \circledR x[]$

However, we should next apply ($\forall$ R) to put $\forall x$ on the right hand side, but this is now blocked by its side condition, because x is free on the left hand side. So, instead, we are forced to first apply ($\circledR \vdash$) with a different variable y. The ($\forall$ R) side condition $x \notin fv(y \circledR \forall x.x[])$ then reduces to the side condition $x \neq y$ of the rule:

$\vdash_{(\circledR \vdash)}$ $y \circledR \forall x.x[] \vdash y \circledR x[]$ where $x \neq y$

$\vdash_{(\forall\, R)}$ $y \circledR \forall x.\mathcal{A} \vdash \forall x.y \circledR \mathcal{A}$ where $x \neq y$

Note that, for two free variables x,y, a disjointness side condition is stronger than a schematic side condition: we could have "where $x \neq y$" satisfied by taking x to be literally a different variable from y, but they could both be instantiated to the same name n, thereby violating a "$(x \neq y)$" side condition. □

Instantiation Principle

An instantiation principle follows from the definition of validity.

4-1 Proposition (Instantiation Principle) (Proof in the Appendix)

(1) $\mathit{vld}(\mathcal{A}) \Rightarrow \mathit{vld}(\mathcal{A}\{x \leftarrow n\})$

(2) Let $\mathcal{S}$ be a one-directional sequent. Then: (Inst) $\mathcal{S} \vdash \mathcal{S}\{x \leftarrow n\}$.

□

Substitution Principle

Let $\mathcal{B}\{-\}$ be a formula with a set of formula holes, indicated by $-$, and let $\mathcal{B}\{\mathcal{A}\}$ denote the formula obtained by filling those holes with the formula $\mathcal{A}$, after renaming the bound variables of $\mathcal{B}$ so they do not capture free variables of $\mathcal{A}$. For any mapping $\varphi \in \vartheta \rightarrow \Lambda$, we have $\mathcal{B}\{-\}_\varphi = \mathcal{B}_\varphi\{-\}$ and $\mathcal{B}\{\mathcal{A}\}_\varphi = \mathcal{B}_\varphi\{\mathcal{A}_\varphi\}$.

4-2 Proposition (Substitution Principle) (Proof in the Appendix)

(1) $\mathit{vld}(\mathcal{A}' \Leftrightarrow \mathcal{A}'') \Rightarrow \mathit{vld}(\mathcal{B}\{\mathcal{A}'\} \Leftrightarrow \mathcal{B}\{\mathcal{A}''\})$

(2) (Subst) $\mathcal{A}' \dashv\vdash \mathcal{A}'' \vdash \mathcal{B}\{\mathcal{A}'\} \dashv\vdash \mathcal{B}\{\mathcal{A}''\}$

□

Case Analysis Principle

A case analysis principle is useful for proofs involving equality and inequality; inequalities often occur as side-conditions of primitive and derived rules.

4-3 Definition (Classical Predicates)

A predicate $\mathcal{A}$ is classical iff $\forall \varphi \in fv(\mathcal{A}) \rightarrow \Lambda.\ \{P \parallel P \models \mathcal{A}_\varphi\} \in \{\Pi, \emptyset\}$.

□

Remark. $\mathbf{T}$, $\mathbf{F}$, and $\eta = \mu$ are classical predicates. So is $\mathcal{A}^{\mathbf{F}}$, for any $\mathcal{A}$ (meaning that $\mathcal{A}$ is unsatisfiable). So is the conjunction, disjunction, and negation of classical predicates. □

4-4 Proposition (Case Analysis Principle) (Proof in the Appendix)

(1) Let $\mathcal{A}$ be a classical predicate. Then: $\mathit{vld}(\mathcal{B}\{\mathbf{T}\}) \wedge \mathit{vld}(\mathcal{B}\{\mathbf{F}\}) \Rightarrow \mathit{vld}(\mathcal{B}\{\mathcal{A}\})$.

(2) Let $\mathcal{S}\{-\}$ be a one-directional sequent with a set of formula holes, and $\mathcal{A}$ be a classical predicate. Then: (Case Analysis) $\mathcal{S}\{\mathbf{T}\}; \mathcal{S}\{\mathbf{F}\} \vdash \mathcal{S}\{\mathcal{A}\}$

□

4.1 Propositional Logic

The following is a non-standard presentation of the propositional sequent calculus [25], based on our single-assumption single-conclusion sequents.

4-5 Proposition (Validity: Propositional Logic) (Proof in the Appendix)

(A-L) $\mathcal{A}\wedge(\mathcal{C}\wedge\mathcal{D}) \vdash \mathcal{B} \;\{\!\}\; (\mathcal{A}\wedge\mathcal{C})\wedge\mathcal{D} \vdash \mathcal{B}$
(A-R) $\mathcal{A} \vdash (\mathcal{C}\vee\mathcal{D})\vee\mathcal{B} \;\{\!\}\; \mathcal{A} \vdash \mathcal{C}\vee(\mathcal{D}\vee\mathcal{B})$
(X-L) $\mathcal{A}\wedge\mathcal{C} \vdash \mathcal{B} \;\}\; \mathcal{C}\wedge\mathcal{A} \vdash \mathcal{B}$
(X-R) $\mathcal{A} \vdash \mathcal{C}\vee\mathcal{B} \;\}\; \mathcal{A} \vdash \mathcal{B}\vee\mathcal{C}$
(C-L) $\mathcal{A}\wedge\mathcal{A} \vdash \mathcal{B} \;\}\; \mathcal{A} \vdash \mathcal{B}$
(C-R) $\mathcal{A} \vdash \mathcal{B}\vee\mathcal{B} \;\}\; \mathcal{A} \vdash \mathcal{B}$
(W-L) $\mathcal{A} \vdash \mathcal{B} \;\}\; \mathcal{A}\wedge\mathcal{C} \vdash \mathcal{B}$
(W-R) $\mathcal{A} \vdash \mathcal{B} \;\}\; \mathcal{A} \vdash \mathcal{C}\vee\mathcal{B}$
(Id) $\}\; \mathcal{A} \vdash \mathcal{A}$
(Cut) $\mathcal{A} \vdash \mathcal{C}\vee\mathcal{B};\ \mathcal{A}'\wedge\mathcal{C} \vdash \mathcal{B}' \;\}\; \mathcal{A}\wedge\mathcal{A}' \vdash \mathcal{B}\vee\mathcal{B}'$
(**T**) $\mathcal{A}\wedge\mathbf{T} \vdash \mathcal{B} \;\}\; \mathcal{A} \vdash \mathcal{B}$
(**F**) $\mathcal{A} \vdash \mathbf{F}\vee\mathcal{B} \;\}\; \mathcal{A} \vdash \mathcal{B}$
(¬-L) $\mathcal{A} \vdash \mathcal{C}\vee\mathcal{B} \;\}\; \mathcal{A}\wedge\neg\mathcal{C} \vdash \mathcal{B}$
(¬-R) $\mathcal{A}\wedge\mathcal{C} \vdash \mathcal{B} \;\}\; \mathcal{A} \vdash \neg\mathcal{C}\vee\mathcal{B}$
□

Remark (Propositional Logic). The standard deduction rules of propositional logic, both for the sequent calculus and for natural deduction, are derivable from the rules of Proposition 4-5. As usual, $\mathcal{A} \Rightarrow \mathcal{B}$ can be defined as $\neg\mathcal{A}\vee\mathcal{B}$. □

4.2 Composition

The composition rules apply to our Ambient Calculus but also, for example, to CCS.

4-6 Proposition (Validity: Composition Rules)

(| **0**)	$\}\; \mathcal{A} \mid \mathbf{0} \dashv\vdash \mathcal{A}$	**0** is nothing
(| ¬**0**)	$\}\; \mathcal{A} \mid \neg\mathbf{0} \vdash \neg\mathbf{0}$	if a part is non-**0**, so is the whole
(A |)	$\}\; \mathcal{A} \mid (\mathcal{B} \mid \mathcal{C}) \dashv\vdash (\mathcal{A} \mid \mathcal{B}) \mid \mathcal{C}$	| associativity
(X |)	$\}\; \mathcal{A} \mid \mathcal{B} \vdash \mathcal{B} \mid \mathcal{A}$	| commutativity
(| ⊢)	$\mathcal{A}' \vdash \mathcal{B}';\ \mathcal{A}'' \vdash \mathcal{B}'' \;\}\; \mathcal{A}' \mid \mathcal{A}'' \vdash \mathcal{B}' \mid \mathcal{B}''$	| congruence
(| ∨)	$\}\; (\mathcal{A}\vee\mathcal{B}) \mid \mathcal{C} \vdash \mathcal{A} \mid \mathcal{C} \vee \mathcal{B} \mid \mathcal{C}$	|-∨ distribution
(| ▷)	$\mathcal{A} \mid \mathcal{C} \vdash \mathcal{B} \;\{\!\}\; \mathcal{A} \vdash \mathcal{C}\triangleright\mathcal{B}$	|-▷ adjunction
(| ||)	$\}\; \mathcal{A}' \mid \mathcal{A}'' \wedge \mathcal{B}' \,\|\|\, \mathcal{B}'' \vdash \mathcal{A}' \mid \mathcal{B}'' \vee \mathcal{B}' \mid \mathcal{A}''$	decomposition

□

The converse of \|-∨ distribution, $\mathcal{A} \mid \mathcal{C} \vee \mathcal{B} \mid \mathcal{C} \vdash (\mathcal{A}\vee\mathcal{B}) \mid \mathcal{C}$, is derivable, and so is a \|-∧ distribution rule, $(\mathcal{A}\wedge\mathcal{B}) \mid \mathcal{C} \vdash \mathcal{A} \mid \mathcal{C} \wedge \mathcal{B} \mid \mathcal{C}$. However, the converse of that, namely $\mathcal{A} \mid \mathcal{C} \wedge \mathcal{B} \mid \mathcal{C} \vdash (\mathcal{A}\wedge\mathcal{B}) \mid \mathcal{C}$, is not sound. (Take $\mathcal{A} = n[m[\mathbf{T}]]$, $\mathcal{B} = n[p[\mathbf{T}]]$, $\mathcal{C} = n[\mathbf{T}]$, and $P = n[m[]] \mid n[p[]]$; then $P \models \mathcal{A} \mid \mathcal{C}$ and $P \models \mathcal{B} \mid \mathcal{C}$, but $\neg\, P \models (\mathcal{A}\wedge\mathcal{B}) \mid \mathcal{C}$.) As a consequence, one

cannot always "push | inside $\wedge$" on the left-hand side of a sequent. In particular, after an application of (| $\vdash$) one cannot in general renormalize a sequent to bring $\wedge$'s to the top level.

The decomposition axiom, (| ||), can be used to analyze a composition $\mathcal{A}' \mid \mathcal{A}''$ with respect to arbitrarily chosen $\mathcal{B}'$ and $\mathcal{B}''$. An easy consequence of it is $\neg(\mathcal{A} \mid \mathcal{B}) \vdash (\mathcal{A} \mid \mathbf{T}) \Rightarrow (\mathbf{T} \mid \neg\mathcal{B})$, which means that if a process cannot be decomposed into parts that satisfy $\mathcal{A}$ and $\mathcal{B}$, but can be decomposed in such a way that a part satisfies $\mathcal{A}$, then it can also be decomposed in such a way that a part does not satisfy $\mathcal{B}$. An even simpler consequence is that $\neg(\mathbf{T} \mid \mathcal{B}) \vdash \mathbf{T} \mid \neg\mathcal{B}$, which is one of the few cases in which one can push $\neg$ across |.

The rule (| ||) is particularly intriguing. In the form:

$$\left\{ \begin{array}{l} \mathcal{A}' \mid \mathcal{A}'' \vdash \mathcal{A}' \mid \mathcal{B}'' \vee \mathcal{B}' \mid \mathcal{A}'' \vee \neg\mathcal{B}' \mid \neg\mathcal{B}'' \end{array} \right.$$

it can be used to analyze a composition $\mathcal{A}' \mid \mathcal{A}''$ with respect to arbitrarily chosen $\mathcal{B}'$ and $\mathcal{B}''$. The rule (| $\neg$), given in the Logical Corollaries 10-2, is a simple consequence of this, while the rule (|-E), again given in the next Logical Corollaries 10-2, is an elimination-style formulation of (| ||).

The rule (| $\triangleright$) states that $\mathcal{A} \triangleright \mathcal{B}$ and $\mathcal{A} \mid \mathcal{B}$ are adjuncts. This has a large number of interesting consequences, most of them deriving from the adjunction along standard lines (see the Appendix).

It is worth pointing out that some composition rules produce interesting interactions between the $\wedge$ and | fragments of the logic. For example, $(\mathcal{A} \mid \mathcal{B}) \wedge \mathbf{0} \vdash \mathcal{A}$ is derivable using (| ||) and (| $\neg\mathbf{0}$).

Axiom naming conventions. This is not a strict rule, but very often axioms are named by a sequence of connectives corresponding to a top-down path through the formula on the left-hand side of the sequent. Occasionally, different but closely related rules have the same name: this is intentional and causes little ambiguity.

4.3 Locations

The location rules are very specific to the Ambient Calculus.

4-7 Proposition (Validity: Location Rules)

(n[] $\neg\mathbf{0}$)	$\left\{ \begin{array}{l} x[\mathcal{A}] \vdash \neg\mathbf{0} \end{array} \right.$	locations exist
(n[] $\neg$ |)	$\left\{ \begin{array}{l} x[\mathcal{A}] \vdash \neg(\neg\mathbf{0} \mid \neg\mathbf{0}) \end{array} \right.$	locations are not decomposable
(n[] $\vdash$)	$\mathcal{A} \vdash \mathcal{B} \left\{ \begin{array}{l} \\ \end{array} \right. x[\mathcal{A}] \vdash x[\mathcal{B}]$	n[] congruence
(n[] $\wedge$)	$\left\{ \begin{array}{l} x[\mathcal{A}] \wedge x[\mathcal{B}] \vdash x[\mathcal{A} \wedge \mathcal{B}] \end{array} \right.$	n[]-$\wedge$ distribution
(n[] $\vee$)	$\left\{ \begin{array}{l} x[\mathcal{A} \vee \mathcal{B}] \vdash x[\mathcal{A}] \vee x[\mathcal{B}] \end{array} \right.$	n[]-$\vee$ distribution
(n[] @)	$x[\mathcal{A}] \vdash \mathcal{B} \left\{ \begin{array}{l} \\ \end{array} \right. \mathcal{A} \vdash \mathcal{B}@x$	n[]-@ adjunction
($\neg$ @)	$\left\{ \begin{array}{l} \mathcal{A}@x \dashv\vdash \neg((\neg\mathcal{A})@x) \end{array} \right.$	@ is self-dual

□

The rule (n[] @) states that $\mathcal{A}@n$ and $n[\mathcal{A}]$ are adjuncts. Note that (n[] $\vdash$) holds in both directions, and that the inverse directions of (n[] $\wedge$) and (n[] $\vee$) are derivable; hence, the location fragment of the logic is particularly simple to handle. Some consequences are: $x[\mathcal{A}@x] \vdash \mathcal{A}$, and $\mathcal{A} \dashv\vdash x[\mathcal{A}]@x$, and $\neg x[\mathcal{A}] \dashv\vdash \neg x[\mathbf{T}] \vee x[\neg\mathcal{A}]$.

4.4 Time and Space Modalities

Both modal operators, $\Box$ and $\maltese$, obey the rules of S4 modalities; these follow simply from reflexivity and transitivity of $\rightarrow^*$ and $\downarrow^*$.

4-8 Proposition (Validity: $\Box$,$\Diamond$ and $\maltese$,$\varpropto$ are Modal S4)

($\Diamond$)	$\vdash \Diamond\mathcal{A} \dashv\vdash \neg\Box\neg\mathcal{A}$	($\varpropto$)	$\vdash \varpropto\mathcal{A} \dashv\vdash \neg\maltese\neg\mathcal{A}$
($\Box$ K)	$\vdash \Box(\mathcal{A} \Rightarrow \mathcal{B}) \vdash \Box\mathcal{A} \Rightarrow \Box\mathcal{B}$	($\maltese$ K)	$\vdash \maltese(\mathcal{A} \Rightarrow \mathcal{B}) \vdash \maltese\mathcal{A} \Rightarrow \maltese\mathcal{B}$
($\Box$ T)	$\vdash \Box\mathcal{A} \vdash \mathcal{A}$	($\maltese$ T)	$\vdash \maltese\mathcal{A} \vdash \mathcal{A}$
($\Box$ 4)	$\vdash \Box\mathcal{A} \vdash \Box\Box\mathcal{A}$	($\maltese$ 4)	$\vdash \maltese\mathcal{A} \vdash \maltese\maltese\mathcal{A}$
($\Box$ **T**)	$\vdash \mathbf{T} \vdash \Box\mathbf{T}$	($\maltese$ **T**)	$\vdash \mathbf{T} \vdash \maltese\mathbf{T}$
($\Box \vdash$)	$\mathcal{A} \vdash \mathcal{B} \vdash \Box\mathcal{A} \vdash \Box\mathcal{B}$	($\maltese \vdash$)	$\mathcal{A} \vdash \mathcal{B} \vdash \maltese\mathcal{A} \vdash \maltese\mathcal{B}$

□

However, these operators are not S5 modalities:

$\neg$ ***vld***$(\Diamond\mathcal{A} \Rightarrow \Box\Diamond\mathcal{A})$

$\neg$ ***vld***$(\varpropto\mathcal{A} \Rightarrow \maltese\varpropto\mathcal{A})$

Namely: If $\mathcal{A}$ may happen along some execution branch, it is not necessarily true that it may happen starting from every execution sub-branch. If $\mathcal{A}$ holds in some sublocation, it is not necessarily true that it holds in some sublocation of every sublocation.

The two modalities permute in one direction (somewhere sometime implies sometime somewhere), but the other direction is not sound. (Consider $P = (\mathit{open}\ n.\ m[p[]]) \mid n[]$. Then $P \vDash \Diamond\varpropto p[\mathbf{0}]$, but $P \nvDash \varpropto\Diamond p[\mathbf{0}]$). The modalities differ prominently in the way they distribute over compositions and locations.

4-9 Proposition (Validity: Other Properties of Modalities)

($\varpropto\Diamond$)	$\vdash \varpropto\Diamond\mathcal{A} \vdash \Diamond\varpropto\mathcal{A}$
($\Diamond\ n[]$)	$\vdash n[\Diamond\mathcal{A}] \vdash \Diamond n[\mathcal{A}]$
($\Diamond \mid$)	$\vdash \Diamond\mathcal{A} \mid \Diamond\mathcal{B} \vdash \Diamond(\mathcal{A} \mid \mathcal{B})$
($\varpropto\ n[]$)	$\vdash n[\varpropto\mathcal{A}] \vdash \varpropto\mathcal{A}$
($\varpropto \mid$)	$\vdash \varpropto\mathcal{A} \mid \mathcal{B} \vdash \varpropto(\mathcal{A} \mid \mathbf{T})$

□

4.5 Satisfiability

Validity and satisfiability can be reflected into the logic by means of the $\mathcal{A}^{\mathbf{F}}$ operator (here we use $\mathcal{A}^{\neg}$ for $\neg\mathcal{A}$):

$\mathcal{A}^{\mathbf{F}}$	$\triangleq\ \mathcal{A}\triangleright\mathbf{F}$	$P \vDash \mathcal{A}^{\mathbf{F}}$ iff $\forall P'{:}\Pi.\ \neg P' \vDash \mathcal{A}$	$\mathcal{A}$ is unsatisfiable
$\mathit{Vld}\,\mathcal{A}$	$\triangleq\ \mathcal{A}^{\neg\mathbf{F}}$	$P \vDash \mathit{Vld}\,\mathcal{A}$ iff $\forall P'{:}\Pi.\ P' \vDash \mathcal{A}$	$\mathcal{A}$ is valid
$\mathit{Sat}\,\mathcal{A}$	$\triangleq\ \mathcal{A}^{\mathbf{F}\neg}$	$P \vDash \mathit{Sat}\,\mathcal{A}$ iff $\exists P'{:}\Pi.\ P' \vDash \mathcal{A}$	$\mathcal{A}$ is satisfiable

From the definitions of $\triangleright$ and $\mathbf{F}$, we obtain that $P \vDash \mathcal{A}^{\mathbf{F}} \Leftrightarrow (\forall P'{:}\Pi.\ P' \vDash \mathcal{A} \Rightarrow P \mid P' \vDash \mathbf{F}) \Leftrightarrow (\forall P'{:}\Pi.\ \neg P' \vDash \mathcal{A})$, i.e., iff $\mathcal{A}$ is unsatisfiable independently of P.

One of the main properties of $\mathcal{A}^{\mathbf{F}}$ is that $\mathcal{A} \mid \mathcal{A}^{\mathbf{F}} \vdash \mathbf{F}$, by ($\triangleright \mid$). That is, $\mathcal{A}$ cannot be both

satisfiable and unsatisfiable. In addition we obtain, from the model, the following rules, from which it is possible to show within the logic that *Vld* and *Sat* obey the rules of S5 modal operators (see Logical Corollaries 10-10).

4-10 Proposition (Validity: Satisfiability Rules)

($\triangleright\mathbf{F}\ \neg$) $\quad \nmid\ \mathcal{A}^{\mathbf{F}} \vdash \mathcal{A}^{\neg}$

($\neg\ \triangleright\mathbf{F}$) $\quad \nmid\ \mathcal{A}^{\mathbf{F}\neg} \vdash \mathcal{A}^{\mathbf{FF}}$

□

Note: $\mathcal{A}^{\neg\mathbf{F}} \vdash \mathcal{A}^{\mathbf{F}\neg}$ is derivable, while $\mathcal{A}^{\mathbf{F}\neg} \vdash \mathcal{A}^{\neg\mathbf{F}}$ and $\mathcal{A}^{\neg} \vdash \mathcal{A}^{\mathbf{F}}$ are not valid.

4.6 Quantifiers

4-11 Proposition (Validity: Quantifiers)

($\forall$-L) $\quad \mathcal{A}\{x\leftarrow\eta\} \vdash \mathcal{B} \ \nmid\ \forall x.\mathcal{A} \vdash \mathcal{B}$ $\quad$ where η is a name or a variable

($\forall$-R) $\quad \mathcal{A} \vdash \mathcal{B} \ \nmid\ \mathcal{A} \vdash \forall x.\mathcal{B}$ $\quad$ where $x \notin fv(\mathcal{A})$

□

As an example, $\Diamond\forall x.\neg(x[\mathbf{T}]^{\exists})$ is the formula for "somewhere there are no ambients". Since there are no infinite spatial paths $P_1 \downarrow P_2 \downarrow P_3 \downarrow$ ..., we can show in the model that this formula is valid. On the other hand, its temporal dual, "sometime there are no ambients", $\Box\forall x.\neg(x[\mathbf{T}]^{\exists})$, is invalid; for instance, it is not satisfied by $n[]$.

For future reference (Corollary 6-6, ($\not\parallel$)):
Note that $(\exists x.\mathcal{A}) \mid (\exists x.\mathcal{B}) \vdash \exists x.(\mathcal{A} \mid \mathcal{B})$ does not hold.
Ex.: $n[] \mid m[m[]] \vDash (\exists x.x[]) \mid (\exists x.x[x[]])$ but $n[] \mid m[m[]] \nvDash \exists x.(x[] \mid x[x[]])$.
Note that $\forall x.(\mathcal{A} \mid \mathcal{B}) \vdash (\forall x.\mathcal{A}) \mid (\forall x.\mathcal{B})$ does not hold.
Ex.: $n[] \mid m[] \vDash \forall x.\ (x{=}n \Rightarrow n[]) \wedge (x{=}m \Rightarrow m[]) \mid (x{=}n \Rightarrow m[]) \wedge (x{=}m \Rightarrow n[])$
but $n[] \mid m[] \nvDash (\forall x.(x{=}n \Rightarrow n[]) \wedge (x{=}m \Rightarrow m[])) \mid (\forall x.(x{=}n \Rightarrow m[]) \wedge (x{=}m \Rightarrow n[]))$

4.7 Name Equality

It is possible to encode name equality within the logic in terms of location adjuncts.

Encoding Name Equality

$\eta = \mu \quad \triangleq \quad \eta[\mathbf{T}]@\mu$

4-12 Proposition (Name Equality)

$\forall\varphi{\in}fv(\eta,\mu){\to}\Lambda.\ \forall P{:}\Pi.\ (P \vDash (\eta = \mu)_{\varphi} \ \Leftrightarrow\ \varphi(\eta) = \varphi(\mu))$

Proof

If $\varphi(\eta) = \varphi(\mu)$, then $\varphi(\mu)[P] \vDash \varphi(\eta)[\mathbf{T}]$, that is $P \vDash \varphi(\eta)[\mathbf{T}]@\varphi(\mu)$, or $P \vDash (\eta[\mathbf{T}]@\mu)_{\varphi}$, or $P \vDash (\eta = \mu)_{\varphi}$.

Conversely, assume that $P \models (\eta = \mu)_\varphi$, that is $P \models \varphi(\eta)[\mathbf{T}]@\varphi(\mu)$. By definition of satisfaction, $P \models \varphi(\eta)[\mathbf{T}]@\varphi(\mu)$ iff $\varphi(\mu)[P] \models \varphi(\eta)[\mathbf{T}]$ iff $\exists P'{:}\Pi.\ \varphi(\mu)[P] \equiv \varphi(\eta)[P']$. But $\varphi(\mu)[P] \equiv \varphi(\eta)[P']$, by Lemmas 2-1, implies $\varphi(\mu) = \varphi(\eta)$.

□

In order to prove statements involving name equality, it is often useful to reason by cases on equality or inequality, using the case analysis principle.

Example:

$\eta{=}\mu \mid \mathcal{B} \vdash \eta{=}\mu$

by case analysis, $\mathbf{T} \mid \mathcal{B} \vdash \mathbf{T}$ by $(\mathcal{A}\mathbf{T})$, and $\mathbf{F} \mid \mathcal{B} \vdash \mathbf{F}$ by $(\mid \mathbf{F})$.

Example:

$(\eta{=}\mu \wedge \mathcal{A}) \mid \mathcal{B} \dashv\vdash \eta{=}\mu \wedge (\mathcal{A} \mid \mathcal{B})$

by case analysis:

$(\mathbf{T} \wedge \mathcal{A}) \mid \mathcal{B} \dashv\vdash \mathcal{A} \mid \mathcal{B} \dashv\vdash \mathbf{T} \wedge (\mathcal{A} \mid \mathcal{B})$;

$(\mathbf{F} \wedge \mathcal{A}) \mid \mathcal{B} \dashv\vdash \mathbf{F} \mid \mathcal{B} \dashv\vdash \mathbf{F} \dashv\vdash \mathbf{F} \wedge (\mathcal{A} \mid \mathcal{B})$.

4.8 Logical Properties of Type Systems

The logic can be used to express properties guaranteed by type systems.

Consider the system of locking and mobility types for the Ambient Calculus [10], recast for the calculus of this paper. The assumption $p{:}Amb^\bullet[S]$ ensures that ambients named p cannot be dissolved by an *open*. We can prove that if $E, p{:}Amb^\bullet[S], E' \vdash P : T$, then $P \models \Box(\Diamond(p[\mathbf{T}]^\exists) \Rightarrow \Box\Diamond(p[\mathbf{T}]^\exists))$. This expresses that in a well-typed process, once an ambient named p somewhere comes into being, ever after there will somewhere be an ambient named p.

Moreover, the assumption $q{:}Amb^\bullet[{}^{\curlyvee}S']$ ensures that ambients named q cannot be moved by *in* or *out* capabilities, nor dissolved by an *open*. We can prove that if $E, p{:}Amb^\bullet[S], q{:}Amb^\bullet[{}^{\curlyvee}S'], E' \vdash P : T$, then $P \models \Box(\Diamond(p[q[\mathbf{T}]^\exists]^\exists) \Rightarrow \Box\Diamond(p[q[\mathbf{T}]^\exists]^\exists))$. This expresses that in a well-typed process, once an ambient named q is somewhere a child of p, ever after there will somewhere be a q child of p.

4.9 Example Derivations

In this section we give examples of derivations that have some intuitive meaning and that can be carried out within the logic. We keep the derivations semi-formal. Some of the rules mentioned below are derived, and can be found in the Appendix.

Shopper and Thief

We use the laws of $\Diamond$, $\mid$, and $\triangleright$, to analyze the consequences of composing two logical specifications. The specifications describe two subsystems: a *Shopper* and a *Thief*, and focus on what happens to the shopper's wallet. The wallet is described simply by the formula *Wallet*[**T**], leaving the contents of the wallet unspecified. The absence of a wallet in a given location is described by the formula *NoWallet*, defined as $\neg(Wallet[\mathbf{T}] \mid \mathbf{T})$, meaning that it

is not possible to decompose the current location into a part containing a wallet and some other part.

A thief is somebody who, in the direct presence of a wallet, can make the wallet disappear. Its specification is $\mathit{Wallet}[\mathbf{T}] \triangleright \Diamond \mathit{NoWallet}$, and its implementation in the Ambient Calculus could simply be given by *open Wallet*.

A shopper is, initially, a person with a wallet (a *Looker*) who is later likely to become a *Buyer*. A buyer is a person who has pulled out the wallet, presumably to buy something. When a wallet has been pulled out, it becomes vulnerable to a nearby thief.

In the following derivation, we show that the interaction of a shopper with a thief (possibly in some larger context) may result in a *CrimeScene*, which is a situation in which the shopper has no wallet, and also there is no wallet to be found nearby.

$$\begin{aligned}
&\mathit{NoWallet} \triangleq \neg(\mathit{Wallet}[\mathbf{T}] \mid \mathbf{T}) \\
&\mathit{Looker} \triangleq \mathit{Person}[\mathit{Wallet}[\mathbf{T}] \mid \mathbf{T}] \\
&\mathit{Buyer} \triangleq \mathit{Person}[\mathit{NoWallet}] \mid \mathit{Wallet}[\mathbf{T}] \\
&\mathit{Shopper} \triangleq \mathit{Looker} \wedge \Diamond \mathit{Buyer} \\
&\mathit{Thief} \triangleq \mathit{Wallet}[\mathbf{T}] \triangleright \Diamond \mathit{NoWallet} \\
&\mathit{CrimeScene} \triangleq \mathit{Person}[\mathit{NoWallet}] \mid \mathit{NoWallet}
\end{aligned}$$

We begin with the system $\mathit{Buyer} \mid \mathit{Thief}$; using the rules $(\mid \triangleright)$ and $(\mid \vdash)$ we obtain:

$$\begin{aligned}
&\mathit{Buyer} \mid \mathit{Thief} \\
&\quad = \mathit{Person}[\mathit{NoWallet}] \mid \mathit{Wallet}[\mathbf{T}] \mid (\mathit{Wallet}[\mathbf{T}] \triangleright \Diamond \mathit{NoWallet}) \\
&\quad \vdash \mathit{Person}[\mathit{NoWallet}] \mid \Diamond \mathit{NoWallet}
\end{aligned}$$

From the rules $(\Diamond \mathrm{T}) \nmid \mathcal{A} \vdash \Diamond \mathcal{A}$, (Id), and $(\mid \vdash)$ we obtain, in general, $\mathcal{A} \mid (\Diamond \mathcal{B}) \vdash (\Diamond \mathcal{A}) \mid (\Diamond \mathcal{B})$. Then, by $(\Diamond \mid) \nmid (\Diamond \mathcal{A}) \mid (\Diamond \mathcal{B}) \vdash \Diamond(\mathcal{A} \mid \mathcal{B})$ and transitivity (derivable from (Cut)) we obtain $\mathcal{A} \mid (\Diamond \mathcal{B}) \vdash \Diamond(\mathcal{A} \mid \mathcal{B})$. Using this fact in our example we obtain, by transitivity:

$$\begin{aligned}
&\mathit{Buyer} \mid \mathit{Thief} \vdash \Diamond(\mathit{Person}[\mathit{NoWallet}] \mid \mathit{NoWallet}) \\
&\quad = \Diamond \mathit{CrimeScene}
\end{aligned}$$

Using the rules $(\Diamond \vdash)$ $\mathcal{A} \vdash \mathcal{B} \nmid \Diamond \mathcal{A} \vdash \Diamond \mathcal{B}$, and $(\Diamond\ 4) \nmid \Diamond\Diamond \mathcal{A} \vdash \Diamond \mathcal{A}$, we derive:

$$\begin{aligned}
&\Diamond(\mathit{Buyer} \mid \mathit{Thief}) \vdash \Diamond\Diamond \mathit{CrimeScene} \\
&\Diamond\Diamond \mathit{CrimeScene} \vdash \Diamond \mathit{CrimeScene}
\end{aligned}$$

As before, we can derive $(\Diamond \mathcal{A}) \mid \mathcal{B} \vdash \Diamond(\mathcal{A} \mid \mathcal{B})$; therefore:

$$(\Diamond \mathit{Buyer}) \mid \mathit{Thief} \vdash \Diamond(\mathit{Buyer} \mid \mathit{Thief})$$

and, by transitivity from above:

$$(\Diamond \mathit{Buyer}) \mid \mathit{Thief} \vdash \Diamond \mathit{CrimeScene}$$

then, by weakening (W-L):

$$(\mathit{Looker} \mid \mathit{Thief}) \wedge ((\Diamond \mathit{Buyer}) \mid \mathit{Thief}) \vdash \Diamond \mathit{CrimeScene}$$

Now let's consider the system *Shopper* | *Thief*. By the distribution of $\mid$ over $\wedge$ $(\mid \wedge) \nmid (\mathcal{A} \wedge \mathcal{B}) \mid C \vdash \mathcal{A} \mid C \wedge \mathcal{B} \mid C)$, we have:

$$Shopper \mid Thief = (Looker \wedge \Diamond Buyer) \mid Thief$$
$$\vdash (Looker \mid Thief) \wedge ((\Diamond Buyer) \mid Thief)$$

and finally, by transitivity from above, we obtain:

$$Shopper \mid Thief \vdash \Diamond CrimeScene$$

Irresistible Force

In the next example, we consider the derivation of a logical paradox, showing that a given specification is not implementable. This is the situation of an immovable object meeting an irresistible force. We say a process contains an immovable object *obj*[] if in the presence of any other process, the object will always exist; this is the *Im* property. We say a process is irresistible (with respect to *obj*[]) if it will make any *obj*[] placed next to it eventually disappear; this is the *Ir* property. The pattern $\mathbf{T} \triangleright \mathcal{A}$ is used to say that "in any context" $\mathcal{A}$ holds.

We consider what happens when we place an immovable object next to an irresistible force; this is done by a spatial composition *Im* | *Ir* (as opposed to, e.g., a conjunction of specifications).

Im	$\triangleq$	$\mathbf{T} \triangleright \Box(obj[] \mid \mathbf{T})$	in any context, there will always be an *obj*[]
Ir	$\triangleq$	$\mathbf{T} \triangleright \Box\Diamond\neg(obj[] \mid \mathbf{T})$	in any context, *obj*[] will disappear

We consider *Im* | *Ir*, and we first expand the definition of *Im*, and lift *Ir* to **T** (by $\mathcal{A} \vdash \mathbf{T}$ and congruence). Then we apply (| $\triangleright$), and a trivial property of $\Diamond$.

$Im \mid Ir \vdash (\mathbf{T} \triangleright \Box(obj[] \mid \mathbf{T})) \mid \mathbf{T}$		because $\mathcal{A} \vdash \mathbf{T}$
$\vdash \Box(obj[] \mid \mathbf{T})$		because $(\mathcal{A} \triangleright \mathcal{B}) \mid \mathcal{A} \vdash \mathcal{B}$
$\vdash \Diamond\Box(obj[] \mid \mathbf{T})$		because $\mathcal{A} \vdash \Diamond\mathcal{A}$

We start again with *Im* | *Ir*, and we now expand the definition of *Ir*, and lift *Im* to **T**. Then we apply again (| $\triangleright$), and commute temporal modalities with negation.

$Im \mid Ir \vdash \mathbf{T} \mid (\mathbf{T} \triangleright \Box\Diamond\neg(obj[] \mid \mathbf{T}))$		because $\mathcal{A} \vdash \mathbf{T}$
$\vdash \Box\Diamond\neg(obj[] \mid \mathbf{T})$		because $(\mathcal{A} \triangleright \mathcal{B}) \mid \mathcal{A} \vdash \mathcal{B}$
$\vdash \neg\Diamond\Box(obj[] \mid \mathbf{T})$		because $\Box\neg\mathcal{A} \vdash \neg\Diamond\mathcal{A}$ etc.

We have now reached opposite conclusions from the same premise, and we can derive a contradiction:

Hence: $Im \mid Ir \vdash \mathbf{F}$	because $\mathcal{A} \wedge \neg\mathcal{A} \vdash \mathbf{F}$

This shows that a system satisfying *Im* | *Ir* cannot be implemented. Therefore, without considering any particular process, we know that either *Im* or *Ir* are not implementable in Ambient Calculus. In fact, *Im* is implementable by !*obj*[], therefore *Ir* must be unimplementable.

See Section 7.3 for a simple derivation involving revelation and the hiding quantifier.

5 Revelation

We now study the logical connectives $\eta \circledR \mathcal{A}$ (*revelation*), and $\mathcal{A} \oslash \eta$ (*revelation adjunct* or *hiding*). These connectives make assertions about restricted names that occur at the process level.

5.1 Satisfaction

The formula $\eta \circledR \mathcal{A}$ is used to *reveal* a restricted name; it is read "reveal η then $\mathcal{A}$", where η is either a name (n) or the occurrence of a variable (x) that denotes a name. A process P satisfies the formula $n \circledR \mathcal{A}$ if it is possible to pull a restricted name occurring in P to the top and rename it n, and then strip off the restriction to leave a residual process that satisfies $\mathcal{A}$.

We cannot rename a top-level restricted name of P to n if n is already free in P. Therefore, a revelation formula provides a way of testing for the free names of the underlying process P, as we discuss below.

The inverse (technically, the adjunct) of revelation is called *hiding*: $\mathcal{A} \oslash \eta$, which is read "hide η then $\mathcal{A}$". A process P satisfies the formula $\mathcal{A} \oslash n$ if $(\nu n)P$ satisfies $\mathcal{A}$, that is, if it is possible to hide n in P and then satisfy $\mathcal{A}$. The satisfaction relation $P \models \mathcal{A}$ for revelation and hiding is repeated below:

Satisfaction for Revelation and Hiding

$$P \models n \circledR \mathcal{A} \triangleq \exists P' \in \Pi.\ P \equiv (\nu n)P' \wedge P' \models \mathcal{A}$$
$$P \models \mathcal{A} \oslash n \triangleq (\nu n)P \models \mathcal{A}$$

Here are some simple examples:

$(\nu n)n[] \models n \circledR \mathbf{T}$	because $n[] \models \mathbf{T}$
$\mathbf{0} \models n \circledR \mathbf{T}$	because $\mathbf{0} \equiv (\nu n)\mathbf{0}$ and $\mathbf{0} \models \mathbf{T}$
$\neg\, n[] \models n \circledR \mathbf{T}$	because there is no process $(\nu n)P' \equiv n[]$
$(\nu m)m[] \models n \circledR n[]$	because $(\nu m)m[] \equiv (\nu n)n[]$ and $n[] \models n[]$
$m[] \models (n \circledR n[]) \oslash m$	because $(\nu m)m[] \models n \circledR n[]$

Revelation gives us a way to talk about the free (or "known") names of a process. This can be embodied in a derived operator $\copyright n$, satisfied by a process P iff $n \in fn(P)$. This and other derived connectives have been already discussed in Section 3.3.

5.2 Rules

Before giving our set of primitive rules of revelation and hiding, we discuss the most interesting properties of $\circledR$ and $\oslash$ that are derived in this section. In order to emphasize some symmetries, we use here a combination of primitive and derived rules,

First, the cancellation and swapping properties of double restriction, $(\nu n)(\nu n)P \equiv (\nu n)P$ and $(\nu n)(\nu m)P \equiv (\nu m)(\nu n)P$, are inherited by both $\circledR$ and $\oslash$:

$n®n®\mathcal{A} \dashv\vdash n®\mathcal{A}$
$\mathcal{A}⦸n⦸n \dashv\vdash \mathcal{A}⦸n$
$n®m®\mathcal{A} \vdash m®n®\mathcal{A}$
$\mathcal{A}⦸m⦸n \vdash \mathcal{A}⦸n⦸m$

Next, consider the combinations:

$n®(\mathcal{A}⦸n)$
$(n®\mathcal{A})⦸n$

We see easily that $P \models n®(\mathcal{A}⦸n)$ means that $P \models \mathcal{A}$ and that $n \notin fn(P)$, where $n \notin fn(P)$ can be written also as $P \models n®\mathbf{T}$. Instead, $P \models (n®\mathcal{A})⦸n$ means that, although P may not satisfy $\mathcal{A}$, if we hide n in P we obtain something where we can reveal n and satisfy $\mathcal{A}$. For example, $(\nu m)n[m[]] \not\models m®m[n[]]$, but $(\nu m)n[m[]] \models (n®m®m[n[]])⦸n$, because $(\nu n)(\nu m)n[m[]] \equiv (\nu n)(\nu m)m[n[]] \models n®m®m[n[]]$. In other words, $P \models (n®\mathcal{A})⦸n$ means that we can satisfy $\mathcal{A}$ by hiding the name n of P, and revealing a possibly different restricted name of P as n. We obtain the properties:

$n®(\mathcal{A}⦸n) \dashv\vdash \mathcal{A} \wedge n®\mathbf{T}$

$n®(\mathcal{A}⦸n) \vdash \mathcal{A}$ $\qquad$ $\mathcal{A} \vdash (n®\mathcal{A})⦸n$
$n®(\mathcal{A}⦸n) \vdash \mathcal{A}⦸n$ $\qquad$ $\mathcal{A}⦸n \vdash (n®\mathcal{A})⦸n$
$n®(\mathcal{A}⦸n) \vdash n®\mathcal{A}$ $\qquad$ $n®\mathcal{A} \vdash (n®\mathcal{A})⦸n$

The interactions of ® and ⦸ with | are the most interesting, and the most complex. There are basically three distribution rules: distribution of ® over | in both directions (with a constraint), unrestricted distribution of ⦸ over | in one direction, and distribution of $n®((-)⦸n)$ over | in both directions.

$n®(\mathcal{A} \mid n®\mathcal{B}) \dashv\vdash n®\mathcal{A} \mid n®\mathcal{B}$
$(\mathcal{A} \mid \mathcal{B})⦸n \vdash \mathcal{A}⦸n \mid \mathcal{B}⦸n$
$n®((\mathcal{A} \mid \mathcal{B})⦸n) \dashv\vdash n®(\mathcal{A}⦸n) \mid n®(\mathcal{B}⦸n)$

The first rule embodies the scope extrusion rule, $(\nu n)(P \mid Q) \equiv ((\nu n)P) \mid Q$ if $n \notin fn(Q)$. This can be seen more clearly if we note that the side condition $n \notin fn(Q)$ is equivalent to $Q \equiv (\nu n)Q$; then the extrusion rule can be written as $(\nu n)(P \mid (\nu n)Q) \equiv ((\nu n)P) \mid ((\nu n)Q)$ with no side condition.

The second rule implies that if $(\nu n)(P \mid Q) \models \mathcal{A} \mid \mathcal{B}$ then it is possible to distribute the restriction so that $(\nu n)P \models \mathcal{A}$ and $(\nu n)Q \models \mathcal{B}$; this is a consequence of Lemma 2-1(9).

The last rule looks mysterious, but has a simple interpretation. According to one of the equivalences above, it can be rewritten as $(\mathcal{A} \mid \mathcal{B}) \wedge n®\mathbf{T} \dashv\vdash (\mathcal{A} \wedge n®\mathbf{T}) \mid (\mathcal{B} \wedge n®\mathbf{T})$; that is, the name n does not occur in a parallel composition iff it does not occur in either component. The right-to-left direction is actually a derivable rule.

A similar set of rules holds for distribution of ® and ⦸ over $n[-]$:

$n®m[\mathcal{A}] \dashv\vdash m[n®\mathcal{A}]$ $\quad (n \neq m)$
$m[\mathcal{A}]⦸n \dashv\vdash m[\mathcal{A}⦸n]$ $\quad (n \neq m)$
$n[\mathcal{A}]⦸n \dashv\vdash \mathbf{F}$
$n®(m[\mathcal{A}]⦸n) \dashv\vdash m[n®(\mathcal{A}⦸n)]$ $\quad (n \neq m)$

The distribution of $n®-$ over $m[-]$ (first rule) holds in both directions as long as $n \neq m$.

The distribution of $-⦸n$ over $m[-]$ (second and third rules) comes in two cases, depending on whether $n{=}m$. In each case, the right-to-left direction is derivable. From $n[\mathbf{T}]⦸n \vdash \mathbf{F}$ we can derive $n®\mathbf{T} \vdash \neg n[\mathbf{T}]$, which means that if a name n does not occur free in a process, the process cannot be a location named n.

The distribution of $n®((-)⦸n)$ over $m[-]$ (fourth rule) is derivable in both directions, from the first two rules. Again, this rule can be rewritten as $m[\mathcal{A}] \wedge n®\mathbf{T} \dashv\vdash m[\mathcal{A} \wedge n®\mathbf{T}]$ $(n \neq m)$; that is, the name n does not occur in a location iff it is distinct from the name of the location and it does not occur inside the location.

Finally, ® and ⦸ commute in one direction:

$$m®(\mathcal{A}⦸n) \vdash (m®\mathcal{A})⦸n$$

We now take the following set of rules as primitive, and we verify their validity in the model. The first group handles double revelation, distribution of ® over $\vee$, congruence of ® with $\vdash$, the adjunction rule connecting ® and ⦸, and the rather curious but very useful fact that $\neg$ commutes with ⦸. The next three groups deal with the interactions of ® and ⦸ with $\mathbf{0}$, $|$, and $n[-]$.

5-1 Proposition (Validity: Revelation Rules)

(®) $\quad \vdash x®x®\mathcal{A} \dashv\vdash x®\mathcal{A}$
(® ®) $\quad \vdash x®y®\mathcal{A} \vdash y®x®\mathcal{A}$
(® $\vee$) $\quad \vdash x®(\mathcal{A} \vee \mathcal{B}) \vdash x®\mathcal{A} \vee x®\mathcal{A}$
(® $\vdash$) $\quad \mathcal{A} \vdash \mathcal{B} \vdash x®\mathcal{A} \vdash x®\mathcal{B}$
(® ⦸) $\quad \eta®\mathcal{A} \vdash \mathcal{B} \{\} \mathcal{A} \vdash \mathcal{B}⦸\eta$
(⦸ $\neg$) $\quad \vdash (\neg\mathcal{A})⦸x \dashv\vdash \neg(\mathcal{A}⦸x)$
(⦸ $\triangleright\mathbf{F}$) $\quad \vdash \mathcal{A}^{\mathbf{F}}⦸x \dashv\vdash \mathcal{A}^{\mathbf{F}}$

(® $\mathbf{0}$) $\quad \vdash x®\mathbf{0} \dashv\vdash \mathbf{0}$
(⦸ $\mathbf{0}$) $\quad \vdash \mathbf{0}⦸x \vdash \mathbf{0}$

(® $|$) $\quad \vdash x®(\mathcal{A} \mid x®\mathcal{B}) \dashv\vdash x®\mathcal{A} \mid x®\mathcal{B}$
(⦸ $|$) $\quad \vdash (\mathcal{A} \mid \mathcal{B})⦸x \vdash \mathcal{A}⦸x \mid \mathcal{B}⦸x$
(® ⦸ $|$) $\quad \vdash x®((\mathcal{A} \mid \mathcal{B})⦸x) \vdash x®(\mathcal{A}⦸x) \mid x®(\mathcal{B}⦸x)$

(® $n[]$) $\quad \vdash x®y[\mathcal{A}] \dashv\vdash y[x®\mathcal{A}]$ $\quad (x \neq y)$
(⦸ $n[]$) $\quad \vdash y[\mathcal{A}]⦸x \vdash y[\mathcal{A}⦸x]$ $\quad (x \neq y)$
(⦸ $n[]$) $\quad \vdash x[\mathcal{A}]⦸x \vdash \mathbf{F}$
□

Remark. The converse of $(\oslash \mid)$ fails. Consider $\mathcal{A}\oslash n \mid \mathcal{B}\oslash n \vdash (\mathcal{A} \mid \mathcal{B})\oslash n$. We have $n[] \mid n[] \models (n\circledR n[])\oslash n \mid (n\circledR n[])\oslash n$, but $n[] \mid n[] \not\models (n\circledR n[] \mid n\circledR n[])\oslash n$. □

Remark. $n\circledR\mathcal{A} \wedge n\circledR\mathcal{B} \vdash n\circledR(\mathcal{A} \wedge \mathcal{B})$ fails (the converse is derivable). We have $(\nu n)(\nu n')n[n'[]] \models n\circledR(n'\circledR n[n'[\mathbf{0}]]) \wedge n\circledR(n'\circledR n'[n[\mathbf{0}]])$, but $(\nu n)(\nu n')n[n'[]] \not\models n\circledR(n'\circledR n[n'[\mathbf{0}]] \wedge n'\circledR n'[n[\mathbf{0}]])$. □

Remark. $n\circledR\mathcal{A} \mid n\circledR\mathcal{B} \dashv\vdash n\circledR(\mathcal{A} \mid \mathcal{B})$ fails in both directions. We have $(\nu n)(n[] \mid n[]) \models n\circledR(n[] \mid n[])$, but $(\nu n)(n[] \mid n[]) \not\models n\circledR n[] \mid n\circledR n[]$. We have $(\nu n)n[] \mid (\nu n)n[] \models n\circledR n[] \mid n\circledR n[]$, but $(\nu n)n[] \mid (\nu n)n[] \not\models n\circledR(n[] \mid n[])$. □

Remark. $\not\vdash (x\circledR\mathcal{A})\oslash y \vdash x\circledR(\mathcal{A}\oslash y)$ $(x \neq y)$ fails. (The converse is derivable without side condition: see $(\oslash \circledR \neq)$ below). For $m \neq n$, we have $(\nu m)m[] \models n\circledR n[]$; therefore $m[] \models (n\circledR n[])\oslash m$. If the rule holds, we then obtain $m[] \models n\circledR(n[]\oslash m)$. This means that $\exists P' \in \Pi.\ m[] \equiv (\nu n)P' \wedge P' \models n[]\oslash m$; that is, $(\nu n)P' \models n[]$, that is $\exists P'' \in \Pi.\ (\nu n)P' \equiv n[P''] \wedge P'' \equiv \mathbf{0}$, that is $(\nu n)P' \equiv n[]$. Then, from $m[] \equiv (\nu n)P'$ and $(\nu n)P' \equiv n[]$ we obtain $m[] \equiv n[]$: contradiction. □

Remark. The converse of $(\oslash \mathbf{0})$, namely $\mathbf{0} \vdash \mathbf{0}\oslash x$, is derivable from $(\circledR \mathbf{0})$. □

Remark. Note that $n[\mathcal{A}]\oslash n \vdash \mathbf{F}$ is the same as $n[\mathbf{T}] \vdash \neg n\circledR\mathbf{T}$, i.e. if n occurs free then it cannot be revealed. □

From the rules that we have validated in Proposition 5-1, we can derive a large collection of facts by logical deduction, including the ones in Logical Corollaries 10-12.

5.3 Hidden-Name Quantifier: First Attempts

The notion of quantifying over a hidden name is surprising subtle. We begin by discussing some attempts that do not work out, and some criteria that such a quantifier should obey.

An Attempt with Bound Names

A *hidden-name quantifier* should be a construct of the logic that allows us to talk about restricted names in processes. The simplest definition that comes to mind is the following for the hidden-name quantifier $\mathsf{H}n.\mathcal{A}$:

$$\mathsf{H}n.\mathcal{A} = \mathsf{H}m.\mathcal{A}\{n \leftarrow m\} \quad \text{if } m \notin fn(\mathcal{A}) \qquad (\alpha\text{-conversion})$$
$$P \models \mathsf{H}n.\mathcal{A} \quad \text{iff} \quad \exists P' \in \Pi.\ P \equiv (\nu n)P' \wedge P' \models \mathcal{A} \qquad (\text{satisfaction})$$

The α-conversion property says that n is a bound name in $\mathsf{H}n.\mathcal{A}$. The definition of satisfaction for $\mathsf{H}n.\mathcal{A}$ is identical to $n\circledR\mathcal{A}$, except for the fact that n is bound.

Unfortunately, there is a problem. Start with the valid assertion $p[] \models \neg n[]$. From the definition above we obtain that $(\nu n)p[] \models \mathsf{H}n.\neg n[]$. By α-conversion we have that $\mathsf{H}n.\neg n[] = \mathsf{H}p.\neg p[]$. So, we would expect that $(\nu n)p[] \models \mathsf{H}p.\neg p[]$. However, this fails, because it is not possible to find a P' such that $(\nu n)p[] \equiv (\nu p)P'$, by Lemma 2-1(1).

Therefore, α-conversion of $\mathsf{H}n.\mathcal{A}$ is not a valid equivalence in the logic, which basically contradicts the notion that n is a bound name in $\mathsf{H}n.\mathcal{A}$.

Moreover, it does not seem possible to simply accept the fact that α-conversion for $\mathsf{H}n.\mathcal{A}$ is not valid. Consider a formula such as $\forall x.\mathsf{H}n.\neg n[x[]]$. Because of the standard rules

for universal quantification, it is possible to instantiate x with the name n. To avoid a name capture, we would have to α-convert $\mathsf{H}n.\neg n[x[]]$ to $\mathsf{H}n'.\neg n'[x[]]$. But if α-conversion is not generally valid for $\mathsf{H}n.\mathcal{A}$, then universal instantiation is also not generally valid.

The basic problem here seems to be the notion of binding names in formulas, so we look next for hidden-name quantifiers that bind variables.

A Discriminating Property

We might now try to define a formula $\mathsf{H}x.\mathcal{B}$ to mean, informally, that "for hidden name x" (hidden in the underlying process), $\mathcal{B}$ holds. The intention is that there should be some correspondence between the binder $\mathsf{H}x$ in the formula, and a binder (νn) in a process that satisfies the formula. There are several plausible definitions. To discriminate between them, we are going to require the following property, (νx-proper), for any candidate definition of $\mathsf{H}x.\mathcal{B}$:

5-2 Property (νx-proper)

For all $n\in\Lambda$, $x\in\vartheta$, $P\in\Pi$, and closed $\mathcal{A}\in\Phi$:

$$n\notin fn(P) \wedge P \vDash \mathsf{H}x.(\mathcal{A}\{n\leftarrow x\}) \quad \Leftrightarrow \quad \exists P'\in\Pi.\ P \equiv (\nu n)P' \wedge P' \vDash \mathcal{A}.$$

Corollary: $P' \vDash \mathcal{A} \ \Rightarrow\ (\nu n)P' \vDash \mathsf{H}x.(\mathcal{A}\{n\leftarrow x\})$.

□

This property can be rewritten in logical form as $n\circledR\mathbf{T} \wedge \mathsf{H}x.(\mathcal{A}\{n\leftarrow x\}) \dashv\vdash n\circledR\mathcal{A}$, for all n.

Assuming (νx-proper) holds, we can already obtain, for example:

$$n[] \vDash n[] \ \Rightarrow\ (\nu n)n[] \vDash \mathsf{H}x.x[]$$
$$p[] \vDash p[] \ \Rightarrow\ (\nu n)p[] \vDash \mathsf{H}x.p[]$$

Remark. It is natural to first consider the simpler discriminating property:

$$(\nu n)P' \vDash \mathsf{H}x.(\mathcal{A}\{n\leftarrow x\}) \quad \Leftrightarrow \quad P' \vDash \mathcal{A} \qquad (\nu x\text{-}1)$$

The $\Leftarrow$ direction is equivalent to (νx-proper$\Leftarrow$). However, the $\Rightarrow$ direction is inconsistent with the fundamental Lemma 3-1. Start with $n[] \vDash n[]$. By (νx-1$\Leftarrow$) we obtain $(\nu n)n[] \vDash \mathsf{H}x.x[]$. Since $(\nu n)n[] \equiv (\nu n)(\nu n)n[]$, by Lemma 3-1 we obtain that $(\nu n)(\nu n)n[] \vDash \mathsf{H}x.x[]$. Then, by ($\nu x-1\Rightarrow$) we obtain $(\nu n)n[] \vDash n[]$, that is $(\nu n)n[] \equiv n[]$, which is contradictory by Lemma 2-1(1). The problem here is that we cannot expect a $\mathsf{H}x$ in the formula to match any (νn) in the process, but only an appropriate one. Hence the refined statement of (νx-proper).
□

An Attempt with Existentials

As our first candidate for $\mathsf{H}x.\mathcal{A}$, it may seem natural to use the following definition: there exists a name x that can be revealed and such that $\mathcal{A}$ is then satisfied:

$$\mathsf{H}x.\mathcal{A} \ \triangleq\ \exists x.x\circledR\mathcal{A}$$

If there exists an m such that the underlying process satisfies $m\circledR\mathcal{A}$, then m is not free in the process, that is, it is fresh. This matches the idea that x should denote a fresh name.

We obtain, directly from the definitions:

$$P \vDash \mathsf{H}x.\mathcal{A} \quad \text{iff} \quad \exists m \in \Lambda.\ \exists P' \in \Pi.\ P \equiv (\nu m)P' \wedge P' \vDash \mathcal{A}\{x \leftarrow m\}$$

We can verify that property (νx-proper$\Leftarrow$) is satisfied. For any n, start with $\exists P' \in \Pi.\ P \equiv (\nu n)P' \wedge P' \vDash \mathcal{A}$, then $n \notin fn(P)$ by Lemma 2-1(1). By definition of revelation we then have that $P \vDash n\circledR\mathcal{A}$. Since $\mathcal{A}$ is closed, this is the same as $P \vDash (x\circledR\mathcal{A}\{n \leftarrow x\})\{x \leftarrow n\}$. We have shown that $\exists n.\ P \vDash (x\circledR\mathcal{A}\{n \leftarrow x\})\{x \leftarrow n\}$. By definition of existential quantification this means that $P \vDash \exists x.\ x\circledR\mathcal{A}\{n \leftarrow x\}$, that is $P \vDash \mathsf{H}x.\mathcal{A}\{n \leftarrow x\}$.

Unfortunately, there is a serious mismatch between this definition of the hidden-name quantifier and our intuitions, because we also obtain:

$$(\nu n)n[] \vDash \mathsf{H}x.p[] \qquad \text{with } n \neq p$$
$$\text{which means: } \exists m \in \Lambda.\ \exists P' \in \Pi.\ (\nu n)n[] \equiv (\nu m)P' \wedge P' \vDash p[]$$

This assertion holds because we can choose $m = p$ and $P' = p[]$ (where p is the name that happens to be free in the formula). Instead, if m is taken to be any other name, then the assertion always fails. So, although p is "fresh" with respect to the process, we cannot equivalently take any other fresh name (with respect to the process) in order to satisfy the formula. This property seems to contradict the concept of "freshness": the assertion holds essentially because of a name clash, and should be intuitively undesirable.

More cogently, this example shows that the property (νx-proper$\Rightarrow$) fails. We have that $(\nu n)n[] \vDash \mathsf{H}x.p[]\{n \leftarrow x\}$, with $n \notin fn((\nu n)n[])$, so we should show that $\exists P' \in \Pi.\ (\nu n)n[] \equiv (\nu n)P' \wedge P' \vDash p[]$. Now, $P' \vDash p[]$ implies, by definition of $\vDash$ and of structural congruence, that $P' \equiv p[]$. Thus, by Lemma 2-1(1), $fn(P') = \{p\}$ and $fn((\nu n)P') = \{p\}$. However, $fn((\nu n)n[]) = \emptyset$. Hence, $(\nu n)n[] \equiv (\nu n)P'$ is impossible, by Lemma 2-1(1).

In conclusion, the existential definition of the hidden-name quantifier satisfies (νx-proper$\Leftarrow$), but violates (νx-proper$\Rightarrow$) because of clashes with free names in formulas.

An Attempt with Universals

Because of the problems with existentials, it may seem better to adopt a universal definition, so as to rule out accidental satisfactions due to the existence of particular names:

$$\mathsf{H}x.\mathcal{A} \triangleq \forall x.x\circledR\mathcal{A}$$

We obtain, directly from the definitions:

$$P \vDash \mathsf{H}x.\mathcal{A} \quad \text{iff} \quad \forall m \in \Lambda.\ \exists P' \in \Pi.\ P \equiv (\nu m)P' \wedge P' \vDash \mathcal{A}\{x \leftarrow m\}$$

As a sanity check, we obtain the expected $(\nu n)n[] \vDash \mathsf{H}x.x[]$, because for any m there exists a $P' = m[]$ such that $(\nu n)n[] \equiv (\nu m)P'$ and $P' \vDash x[]\{x \leftarrow m\}$.

Moreover, this time $(\nu n)n[] \nvDash \mathsf{H}x.p[]$, because if we choose $m \neq p$ then we should show that there exists a P' such that $(\nu n)n[] \equiv (\nu m)P'$ and $P' \vDash p[]$. But $P' \vDash p[]$ implies that $P' \equiv p[]$, and then we would have $(\nu n)n[] \equiv (\nu m)p[]$, which is impossible (Lemma 2-1(1)).

In fact, we find that (νx-proper$\Rightarrow$) is satisfied in general. Assume $n \notin fn(P)$ and $P \vDash \forall x.x\circledR(\mathcal{A}\{n \leftarrow x\})$, that is $\forall m \in \Lambda.\ \exists P' \in \Pi.\ P \equiv (\nu m)P' \wedge P' \vDash \mathcal{A}\{n \leftarrow x\}\{x \leftarrow m\}$. Then, for $m = n$, we obtain in particular that $\exists P' \in \Pi.\ P \equiv (\nu n)P' \wedge P' \vDash \mathcal{A}$.

Unfortunately, this time property (νx-proper$\Leftarrow$) fails. Consider the valid assertion $p[] \vDash \neg n[]$ with $n \neq p$; by (νx-proper$\Leftarrow$) we would obtain that $(\nu n)p[] \vDash \forall x.x\circledR\neg x[]$. This means

that $\forall m \in \Lambda.\ \exists P' \in \Pi.\ (\nu n)p[] \equiv (\nu m)P' \wedge P' \models \neg m[]$. In particular, for $m = p$, we obtain $\exists P' \in \Pi.\ (\nu n)p[] \equiv (\nu p)P' \wedge P' \models \neg p[]$. But $(\nu n)p[] \equiv (\nu p)P'$ is impossible by Lemma 2-1(1).

In conclusion, the universal definition of the hidden-name quantifier satisfies (νx-proper$\Rightarrow$), but violates (νx-proper$\Leftarrow$) because of clashes with free names in processes.

Two Attempt with Fresh Names

A more refined attempt, then, might involve ruling out the names that are free in the formula and the processes, so that we avoid the problems above. For the simple case where $\mathsf{H}x.\mathcal{A}$ is closed, we would have:

$$\mathsf{H}x.\mathcal{A} \triangleq \exists x.\ x \neq n_1 \wedge \ldots \wedge x \neq n_k \wedge x \circledR \mathbf{T} \wedge x \circledR \mathcal{A}$$
$$\text{where } \{n_1, \ldots, n_k\} = fn(\mathcal{A})$$

This is in fact what we will come to eventually (suitably generalizing to open formulas). It can be read as "there exists a fresh name x (distinct from the free names of $\mathcal{A}$ and the free names of the underlying process) such that a restricted process name can be revealed as x, and then $\mathcal{A}$ can be satisfied". This definition will satisfy (νx-proper).

However, there is another plausible definition:

$$\mathsf{H}x.\mathcal{A} \triangleq \forall x.\ (x \neq n_1 \wedge \ldots \wedge x \neq n_k \wedge x \circledR \mathbf{T}) \Rightarrow x \circledR \mathcal{A}$$
$$\text{where } \{n_1, \ldots, n_k\} = fn(\mathcal{A})$$

which can be read "for every fresh name x (distinct from the free names of $\mathcal{A}$ and the free names of the underlying process), a restricted process name can be revealed as x, and then $\mathcal{A}$ can be satisfied".

These two definitions will turn out to be equivalent (Logical Corollaries 10-14). The equivalence of the existential and universal definitions is at the essence of the notion of "freshness" [23]. Before coming back to $\mathsf{H}x.\mathcal{A}$, we study freshness in the next section, independently of revelation.

6 Fresh-Name Quantifier

In this section we define a formula, $\mathsf{N}x.\mathcal{A}$, with the meaning "for fresh x, $\mathcal{A}$ holds". Here, "fresh" means, informally, distinct from any name that might clash with an existing name.

The set of free (i.e., non-fresh) names that occur in a process or formula is always finite; hence sets of fresh names are always cofinite[1]. If there is a suitable fresh x, then there are infinitely many of them, since a fresh name can be replaced by any other fresh name. Therefore, "freshness" can be expressed formally as the existence of a cofinite set of interchangeable names [23].

We use $Fin(S)$ for the collection of finite subsets of a set S, and (rarely) $CoFin(S)$ for the collection of cofinite subsets of S.

---

1. A cofinite set is the complement of a finite set with respect to an infinite universe, which, in our case, is the countable universe of names Λ.

6.1 The Gabbay-Pitts Property

We would like to obtain the following property for $\nu x.\mathcal{A}$:

$$P \vDash \nu x.\mathcal{A} \quad \Leftrightarrow \quad \exists m \in \Lambda.\ m \notin fn(P,\mathcal{A}) \wedge P \vDash \mathcal{A}\{x \leftarrow m\}$$

That is, $P \vDash \nu x.\mathcal{A}$ iff there exists a fresh name m such that $P \vDash \mathcal{A}\{x \leftarrow m\}$.

This definition is given by existential quantification over fresh names. Remarkably, there is an equivalent definition based on universal quantification. The equivalence of these two definitions is based on a deep property of the logic (Lemma 3-3), and will be used to great effect later. We state the equivalence as follows: there exists a fresh name m such that $P \vDash \mathcal{A}\{x \leftarrow m\}$, if and only if for all fresh names m we have $P \vDash \mathcal{A}\{x \leftarrow m\}$:

6-1 Proposition (Gabbay-Pitts Property)

$\forall P \in \Pi, \mathcal{A} \in \Phi, N \in Fin(\Lambda)$.
$N \supseteq fn(P,\mathcal{A}) \wedge fv(\mathcal{A}) \subseteq \{x\} \Rightarrow$
$(\exists m \in \Lambda.\ m \notin N \wedge P \vDash \mathcal{A}\{x \leftarrow m\}) \quad \Leftrightarrow \quad (\forall m \in \Lambda.\ m \notin N \Rightarrow P \vDash \mathcal{A}\{x \leftarrow m\})$

Proof

Assume $N \supseteq fn(P,\mathcal{A})$ and $fv(\mathcal{A}) \subseteq \{x\}$.

Case $\Leftarrow$) Assume $\forall m \in \Lambda.\ m \notin N \Rightarrow P \vDash \mathcal{A}\{x \leftarrow m\}$. Since N is finite and Λ is infinite, there is a $p \in \Lambda$ such that $p \notin N$. Then, by assumption, $P \vDash \mathcal{A}\{x \leftarrow p\}$. We have shown $(\exists p \in \Lambda.\ p \notin N \wedge P \vDash \mathcal{A}\{x \leftarrow p\})$.

Case $\Rightarrow$) Assume $\exists m \in \Lambda.\ m \notin N \wedge P \vDash \mathcal{A}\{x \leftarrow m\}$; in particular, $m \notin fn(P,\mathcal{A})$. Take any $p \in \Lambda$ and assume $p \notin N$. If $p = m$ we have by assumption that $P \vDash \mathcal{A}\{x \leftarrow p\}$. Otherwise, if $p \neq m$ then $p \notin N \cup \{m\}$; since $fn(P,\mathcal{A}\{x \leftarrow m\}) \subseteq N \cup \{m\}$, we have that $p \notin fn(P,\mathcal{A}\{x \leftarrow m\})$. By applying Lemma 3-3 to the assumption $P \vDash \mathcal{A}\{x \leftarrow m\}$ we obtain $P\{m \leftarrow p\} \vDash \mathcal{A}\{x \leftarrow m\}\{m \leftarrow p\}$; that is, $P \vDash \mathcal{A}\{x \leftarrow p\}$. In both cases, we have shown that $(\forall p \in \Lambda.\ p \notin N \Rightarrow P \vDash \mathcal{A}\{x \leftarrow p\})$.

□

The following corollary gives consequences and alternative formulations of the Gabbay-Pitts property in terms of finite sets of names that include $fn(P,\mathcal{A})$ (or, alternatively, in terms of cofinite sets of names that do not intersect $fn(P,\mathcal{A})$).

6-2 Corollary (Proof in the Appendix)

Assume $P \in \Pi, \mathcal{A} \in \Phi, fv(\mathcal{A}) \subseteq \{x\}$. Then,
$\exists m \in \Lambda.\ m \notin fn(P,\mathcal{A}) \wedge P \vDash \mathcal{A}\{x \leftarrow m\}$
(1) $\Leftrightarrow \quad \exists N \in Fin(\Lambda).\ N \supseteq fn(P,\mathcal{A}) \wedge \exists m \in \Lambda.\ m \notin N \wedge P \vDash \mathcal{A}\{x \leftarrow m\}$
(2) $\Leftrightarrow \quad \exists N \in Fin(\Lambda).\ N \supseteq fn(P,\mathcal{A}) \wedge \forall m \in \Lambda.\ m \notin N \Rightarrow P \vDash \mathcal{A}\{x \leftarrow m\}$
(3) $\Leftrightarrow \quad \forall N \in Fin(\Lambda).\ N \supseteq fn(P,\mathcal{A}) \Rightarrow \exists m \in \Lambda.\ m \notin N \wedge P \vDash \mathcal{A}\{x \leftarrow m\}$
(4) $\Leftrightarrow \quad \forall N \in Fin(\Lambda).\ N \supseteq fn(P,\mathcal{A}) \Rightarrow \forall m \in \Lambda.\ m \notin N \Rightarrow P \vDash \mathcal{A}\{x \leftarrow m\}$
□

6.2 A Gabbay-Pitts Logical Rule

We now want to formulate a Gabbay-Pitts property similar to Proposition 6-1, but expressible within the logic. We are going to use extensively the idiom $x\#N \wedge x\circledR\mathbf{T}$, for a quantified variable x. The first part of this conjunction says that the name x is fresh with respect to a given set of names N that usually includes the set of free names of a formula of interest. The second part says that x is fresh in the "underlying process", because $P \vDash n\circledR\mathbf{T}$ iff $n \notin fn(P)$. For a suitable choice of N, the whole conjunction can be understood as saying that x is "completely fresh", both at the formula and process level, in a given situation.

Notation

- For $N \in Fin(\Lambda \cup \vartheta)$ we define the formula $\eta\#N \triangleq \bigwedge_{\mu \in N}(\eta \neq \mu)$.
  For any P and closed $m\#N$, we have $P \vDash m\#N$ iff $m \notin N$.
- Let $fnv(\mathcal{A}) \triangleq fn(\mathcal{A}) \cup fv(\mathcal{A})$, so that $fnv(\mathcal{A}) \in Fin(\Lambda \cup \vartheta)$

With this understanding, the following proposition states the single rule (schema) that we add to our logic in order to capture "freshness", and establishes its soundness. Note that this rule holds for open formulas.

6-3 Proposition (Validity: Gabbay-Pitts)

(GP) $\vdash \exists x.\, x\#N \wedge x\circledR\mathbf{T} \wedge \mathcal{A} \dashv\vdash \forall x.\, (x\#N \wedge x\circledR\mathbf{T}) \Rightarrow \mathcal{A}$
where $N \in Fin(\Lambda \cup \vartheta)$ and $N \supseteq fnv(\mathcal{A})\text{-}\{x\}$ and $x \notin N$

Proof

Assume $N \supseteq fnv(\mathcal{A})\text{-}\{x\}$ and $x \notin N$. We need to show that the sequent is valid, that is that $\forall \varphi \in (fv(\mathcal{A})\text{-}\{x\}) \to \Lambda$, $P \in \Pi$. $P \vDash (\exists x.\, x\#N \wedge x\circledR\mathbf{T} \wedge \mathcal{A})_\varphi \Leftrightarrow P \vDash (\forall x.\, x\#N \wedge x\circledR\mathbf{T} \Rightarrow \mathcal{A})_\varphi$.

(1) $\vdash \exists x.\, x\#N \wedge x\circledR\mathbf{T} \wedge \mathcal{A} \vdash \forall x.\, x\#N \wedge x\circledR\mathbf{T} \Rightarrow \mathcal{A}$

Take any $\varphi \in (fv(\mathcal{A})\text{-}\{x\}) \to \Lambda$ and $P \in \Pi$, and assume $P \vDash (\exists x.\, x\#N \wedge x\circledR\mathbf{T} \wedge \mathcal{A})_\varphi$. That is, assume $\exists m \in \Lambda.\ m \notin N_\varphi \cup fn(P) \wedge P \vDash \mathcal{A}_\varphi\{x \leftarrow m\}$, where $N_\varphi \cup fn(P) \supseteq fn(P, \mathcal{A}_\varphi)$ and $fv(\mathcal{A}_\varphi) \subseteq \{x\}$. By Proposition 6-1, we obtain $\forall m \in \Lambda.\ m \notin N_\varphi \cup fn(P) \Rightarrow P \vDash \mathcal{A}_\varphi\{x \leftarrow m\}$, that is $P \vDash (\forall x.\, x\#N \wedge x\circledR\mathbf{T} \Rightarrow \mathcal{A})_\varphi$.

(2) $\vdash \forall x.\, x\#N \wedge x\circledR\mathbf{T} \Rightarrow \mathcal{A} \vdash \exists x.\, x\#N \wedge x\circledR\mathbf{T} \wedge \mathcal{A}$

Take any $\varphi \in (fv(\mathcal{A})\text{-}\{x\}) \to \Lambda$ and $P \in \Pi$ and assume $P \vDash (\forall x.\, x\#N \wedge x\circledR\mathbf{T} \Rightarrow \mathcal{A})_\varphi$; that is assume $(\forall m \in \Lambda.\ m \notin N_\varphi \cup fn(P) \Rightarrow P \vDash \mathcal{A}_\varphi\{x \leftarrow m\})$, where $N_\varphi \cup fn(P) \supseteq fn(P, \mathcal{A}_\varphi)$ and $fv(\mathcal{A}_\varphi) \subseteq \{x\}$. By Proposition 6-1, we obtain $\exists m \in \Lambda.\ m \notin N_\varphi \cup fn(P) \wedge P \vDash \mathcal{A}_\varphi\{x \leftarrow m\}$, that is $P \vDash (\exists x.\, x\#N \wedge x\circledR\mathbf{T} \wedge \mathcal{A})_\varphi$.

□

Remark. (GP) gives us a way to prove that $\forall x.\mathcal{A} \vdash \exists x.\mathcal{A}$. This depends on the fact that the set of names is non-empty, and is obviously not derivable from the normal quantifier rules. Take $N = fnv(\mathcal{A})\text{-}\{x\}$. Starting from $\mathcal{A} \vdash \mathcal{A}$, by right weakening and quantifier introduction we obtain $\forall x.\, \mathcal{A} \vdash \forall x.\, x\#N \wedge x\circledR\mathbf{T} \Rightarrow \mathcal{A}$. Again starting from $\mathcal{A} \vdash \mathcal{A}$, by left weakening and

quantifier introduction we obtain $\exists x.\ x\#N \wedge x\mathbin{\circledR}\mathbf{T} \wedge \mathcal{A} \vdash \exists x.\ \mathcal{A}$. By (GP) we have $\forall x.\ x\#N \wedge x\mathbin{\circledR}\mathbf{T} \Rightarrow \mathcal{A} \vdash \exists x.\ x\#N \wedge x\mathbin{\circledR}\mathbf{T} \wedge \mathcal{A}$. Hence, by transitivity we obtain $\forall x.\ \mathcal{A} \vdash \exists x.\ \mathcal{A}$. □

6.3 Fresh-Name Quantifier

Now, we can define quantification over fresh names, $\mathcal{N}x.\mathcal{A}$, as follows:

6-4 Definition (Fresh-Name Quantifier)

$\mathcal{N}x.\mathcal{A} \triangleq \exists x.\ x\#fnv(\mathcal{A})\text{-}\{x\} \wedge x\mathbin{\circledR}\mathbf{T} \wedge \mathcal{A}$

□

Hence $fn(\mathcal{N}x.\mathcal{A}) = fn(\mathcal{A})$ and $fv(\mathcal{N}x.\mathcal{A}) = fv(\mathcal{A})\text{-}\{x\}$.

Note that the right-hand side of this definition depends on the set of free names and variables of $\mathcal{A}$. Therefore, this is not a definition within the logic, but rather a meta-theoretical definition (or abbreviation) that should always be understood in its expanded form. Any general theorem or derived rule involving $\mathcal{N}x.\mathcal{A}$ will in fact be a schematic theorem or rule with respect to the free names and variables of $\mathcal{A}$, in the same way that (GP) is a rule schema.

By (GP) (Proposition 6-3) we have:

$$\mathcal{N}x.\mathcal{A} \dashv\vdash \forall x.\ x\#fnv(\mathcal{A})\text{-}\{x\} \wedge x\mathbin{\circledR}\mathbf{T} \Rightarrow \mathcal{A}$$

In terms of satisfaction, we obtain:

6-5 Lemma ($P \models \mathcal{N}x.\mathcal{A}$)

$P \models \mathcal{N}x.\mathcal{A}$
iff $\exists m{\in}\Lambda.\ m{\notin}fn(P,\mathcal{A}) \wedge P \models \mathcal{A}\{x{\leftarrow}m\}$
iff $\forall m{\in}\Lambda.\ m{\notin}fn(P,\mathcal{A}) \Rightarrow P \models \mathcal{A}\{x{\leftarrow}m\}$

Proof

Note that $P \models \mathcal{N}x.\mathcal{A}$ implies that $fv(\mathcal{N}x.\mathcal{A}) = \emptyset$, that is $fv(\mathcal{A}) \subseteq \{x\}$. Similarly, $P \models \mathcal{A}\{x{\leftarrow}m\}$ implies that $fv(\mathcal{A}\{x{\leftarrow}m\}) = \emptyset$, that is $fv(\mathcal{A}) \subseteq \{x\}$.

By definition of $\mathcal{N}$, $P \models \mathcal{N}x.\mathcal{A}$ iff $P \models \exists x.\ x\#fnv(\mathcal{A})\text{-}\{x\} \wedge x\mathbin{\circledR}\mathbf{T} \wedge \mathcal{A}$. Since $fv(\mathcal{A}) \subseteq \{x\}$, this is equivalent to $P \models \exists x.\ x\#fn(\mathcal{A}) \wedge x\mathbin{\circledR}\mathbf{T} \wedge \mathcal{A}$; that is $\exists m{\in}\Lambda.\ P \models m\#fn(\mathcal{A}) \wedge P \models m\mathbin{\circledR}\mathbf{T} \wedge P \models \mathcal{A}\{x{\leftarrow}m\}$. This is the same as $\exists m{\in}\Lambda.\ m{\notin}fn(\mathcal{A}) \wedge m{\notin}fn(P) \wedge P \models \mathcal{A}\{x{\leftarrow}m\}$.

The second equivalence is obtained by Proposition 6-1.

□

Therefore, $\mathcal{N}x.\mathcal{A}$ can be understood as saying either that there is a fresh name x such that $\mathcal{A}$ holds, or that for any fresh name x we have that $\mathcal{A}$ holds. These formulations are equivalent because of the cofinite nature of sets of fresh names. If there is a suitably fresh x such that $\mathcal{A}$ holds, then any other fresh name will work equally well, so all fresh names will work. Conversely, if for all suitably fresh names $\mathcal{A}$ holds, since any set of fresh names is (cofinite and hence) non-empty, there exists a fresh name for which $\mathcal{A}$ holds.

Remark. The meaning of $\mathcal{N}x.\mathcal{A}$ when $\mathcal{A}$ has free variables other than x is subtle. When we write $\mathcal{N}x.\ ...n...$ we intend x to be fresh w.r.t. any existing name, and in particular n; similar-

ly, when we write $И x.\ ...y...$ we intend x to be fresh with respect to any name denoted by y. Consider $И y.\ И x.\ y{=}x$; this formula should not be valid. In fact, it is contradictory because, by definition, it means, $\exists y.\ y \mathbin{®} \mathbf{T} \wedge \exists x.\ x{\neq}y \wedge x \mathbin{®} \mathbf{T} \wedge y{=}x$. Similarly, $\forall y.\ И x.\ y{=}x$ and $\exists y.\ И x.\ y{=}x$ are contradictory. (Instead, $И x.\ \exists y.\ x{=}y$ is valid.) □

The following rules are now derivable entirely within the logic:

6-6 Logical Corollaries (Fresh-Name Quantifier)

(И ∃)	$\vdash И x.\mathcal{A} \dashv\vdash \exists x.\ x\#N \wedge x \mathbin{®} \mathbf{T} \wedge \mathcal{A}$	where $N \supseteq fnv(\mathcal{A})\text{-}\{x\}$ and $x \notin N$
(И ∀)	$\vdash \forall x.\ x\#N \wedge x \mathbin{®} \mathbf{T} \Rightarrow \mathcal{A} \dashv\vdash И x.\mathcal{A}$	where $N \supseteq fnv(\mathcal{A})\text{-}\{x\}$ and $x \notin N$
(И ¬)	$\vdash \neg И x.\mathcal{A} \dashv\vdash И x.\neg\mathcal{A}$	
(И |)	$\vdash И x.(\mathcal{A} \mid \mathcal{B}) \dashv\vdash (И x.\mathcal{A}) \mid (И x.\mathcal{B})$	
(И ⊢)	$\mathcal{A} \vdash \mathcal{B} \ \vdash И x.\mathcal{A} \vdash И x.\mathcal{B}$	
(И *fv*)	$\vdash И x.\mathcal{A} \dashv\vdash \mathcal{A}$	where $x \notin fv(\mathcal{A})$
(И *n*[])	$\vdash И x.y[\mathcal{A}] \dashv\vdash y[И x.\mathcal{A}]$	where $x{\neq}y$
(И R)	$\mathcal{A} \wedge x\#N \wedge x \mathbin{®} \mathbf{T} \vdash \mathcal{B} \ \vdash \mathcal{A} \vdash И x.\mathcal{B}$	where $N \supseteq fnv(\mathcal{B})\text{-}\{x\}$ and $x \notin N \cup fv(\mathcal{A})$
(И L)	$\mathcal{A} \wedge x\#N \wedge x \mathbin{®} \mathbf{T} \vdash \mathcal{B} \ \vdash И x.\mathcal{A} \vdash \mathcal{B}$	where $N \supseteq fnv(\mathcal{A})\text{-}\{x\}$ and $x \notin N \cup fv(\mathcal{B})$
(И E)	$\mathcal{A} \vdash И x.\mathcal{B};\ \mathcal{B} \wedge x\#N \wedge x \mathbin{®} \mathbf{T} \vdash C \ \vdash \mathcal{A} \vdash C$	where $N \supseteq fnv(\mathcal{B})\text{-}\{x\}$ and $x \notin N \cup fv(C)$

□

Remark. The fresh-name quantifier is "in between" universal and existential quantification ($\forall x.\mathcal{A} \vdash И x.\mathcal{A} \vdash \exists x.\mathcal{A}$, by (И ∀) and (И ∃)). Moreover, it is "right in the middle", since it enjoys many properties of both universal and existential quantification; for example, it is self-dual ($И x.\mathcal{A} \dashv\vdash \neg И x.\neg\mathcal{A}$, by (И ∀), (И ∃), and DeMorgan). □

Remark. We can show that $\forall y.\ \exists x.\ x{\neq}y$. Start from $x{\neq}y \wedge x \mathbin{®} \mathbf{T} \Rightarrow x{\neq}y$, and generalize to $\forall y.\ \forall x.\ x{\neq}y \wedge x \mathbin{®} \mathbf{T} \Rightarrow x{\neq}y$. By (И ∀) this means $\forall y.\ И x.\ x{\neq}y$. By the previous remark, $И x.\ x{\neq}y \vdash \exists x.\ x{\neq}y$, hence by (∀ ⊢) we obtain $\forall y.\ \exists x.\ x{\neq}y$. □

Remark. Of particular interest (and difficulty) is the distribution of И over |, rule (И |):

$$\vdash И x.(\mathcal{A} \mid \mathcal{B}) \dashv\vdash (И x.\mathcal{A}) \mid (И x.\mathcal{B})$$

Distribution over | holds in one direction for universal quantification, in the other direction for existential quantification, and in both directions for fresh-name quantification. This rule can be understood informally as follows (this is a sketch of the formal derivation). In the left-to-right direction we use the existential interpretation of И. Take any P; if $P \models И x.(\mathcal{A} \mid \mathcal{B})$ then there are a fresh name x and processes P',P'' such that $P \equiv P' \mid P''$ and $P' \models \mathcal{A}$ and $P'' \models \mathcal{B}$. Hence, there is a fresh name x such that $P' \models \mathcal{A}$ and again a fresh name x such that $P'' \models \mathcal{B}$; that is, $P' \models И x.\mathcal{A}$ and $P'' \models И x.\mathcal{B}$. Therefore, $P \equiv P' \mid P'' \models (И x.\mathcal{A}) \mid (И x.\mathcal{B})$. In the right-to-left direction we use the universal interpretation of И. Take any P; if $P \models (И x.\mathcal{A}) \mid (И x.\mathcal{B})$ then there are processes P',P'' such that $P \equiv P' \mid P''$ and $P' \models И x.\mathcal{A}$ and $P'' \models И x.\mathcal{B}$. This means that for all names x' fresh in P' and $\mathcal{A}$, we have $P' \models \mathcal{A}\{x{\leftarrow}x'\}$ and for all names x'' fresh in P'' and $\mathcal{B}$, we have $P'' \models \mathcal{B}\{x{\leftarrow}x''\}$. Now, for all names y that are fresh in P', $\mathcal{A}$, P'', $\mathcal{B}$; we have that $P' \models \mathcal{A}\{x{\leftarrow}y\}$ and $P'' \models \mathcal{B}\{x{\leftarrow}y\}$. That is, $P \equiv P' \mid P'' \models И y.(\mathcal{A}\{x{\leftarrow}y\} \mid \mathcal{B}\{x{\leftarrow}y\}) = И x.(\mathcal{A} \mid \mathcal{B})$. □

7 Hidden-Name Quantifier

7.1 Definition

We can finally get back to our original aim of defining a hidden-name quantifier at the logical level. We take:

7-1 Definition (Hidden-Name Quantifier)

$\mathsf{H}x.\mathcal{A} \triangleq \mathsf{И}x.x\circledR\mathcal{A}$

□

Hence $fn(\mathsf{H}x.\mathcal{A}) = fn(\mathcal{A})$ and $fv(\mathsf{H}x.\mathcal{A}) = fv(\mathcal{A})\text{-}\{x\}$. Moreover, by definition of И:

$$\mathsf{H}x.\mathcal{A} = \exists x.\, x\#fnv(\mathcal{A})\text{-}\{x\} \wedge x\circledR\mathbf{T} \wedge x\circledR\mathcal{A}$$

and, because of Logical Corollary 10-12(® ∧), we can simplify this to:

$$\mathsf{H}x.\mathcal{A} \dashv\vdash \exists x.\, x\#fnv(\mathcal{A})\text{-}\{x\} \wedge x\circledR\mathcal{A}$$

In terms of satisfaction, we obtain:

7-2 Lemma ($P \vDash (\nu x)\mathcal{A}$)

$P \vDash \mathsf{H}x.\mathcal{A}$ iff

$\exists m\in\Lambda.\ m\notin fn(P,\mathcal{A}) \wedge \exists P'\in\Pi.\ P \equiv (\nu m)P' \wedge P' \vDash \mathcal{A}\{x\leftarrow m\}$

Proof

Satisfaction is defined for closed formulas, so $\exists x.\, x\#fnv(\mathcal{A})\text{-}\{x\} \wedge x\circledR\mathbf{T} \wedge x\circledR\mathcal{A}$ is the same as $\exists x.\, x\#fn(\mathcal{A}) \wedge x\circledR\mathbf{T} \wedge x\circledR\mathcal{A}$. Then, $P \vDash \exists x.\, x\#fn(\mathcal{A}) \wedge x\circledR\mathbf{T} \wedge x\circledR\mathcal{A}$ means that $\exists m\in\Lambda.\ m\notin fn(\mathcal{A}) \wedge m\notin fn(P) \wedge P \vDash m\circledR\mathcal{A}\{x\leftarrow m\}$, and from definition of satisfaction for ®, we obtain the statement.

□

7.2 Properties

We can now verify that this definition of the hidden-name quantifier fully satisfies the property (νx-proper):

7-3 Proposition (νx-proper)

For all $n\in\Lambda$, $x\in\vartheta$, $P\in\Pi$, and closed $\mathcal{A}\in\Phi$:

$n\notin fn(P) \wedge P \vDash \mathsf{H}x.(\mathcal{A}\{n\leftarrow x\}) \quad\Leftrightarrow\quad \exists P'\in\Pi.\ P \equiv (\nu n)P' \wedge P' \vDash \mathcal{A}$

Proof

Case ⇒)

$P \vDash \mathsf{H}x.(\mathcal{A}\{n\leftarrow x\}) \Rightarrow P \vDash \mathsf{И}x.x\circledR(\mathcal{A}\{n\leftarrow x\})$

$\Rightarrow \exists m\in\Lambda.\ m\notin fn(P,x\circledR(\mathcal{A}\{n\leftarrow x\})) \wedge \exists P'\in\Pi.\ P \equiv (\nu m)P' \wedge P' \vDash \mathcal{A}\{n\leftarrow x\}\{x\leftarrow m\}$

$\Rightarrow \exists m\in\Lambda.\ m\notin fn(P,\mathcal{A}\{n\leftarrow x\}) \wedge \exists P'\in\Pi.\ P \equiv (\nu m)P' \wedge P' \vDash \mathcal{A}\{n\leftarrow m\}$

Suppose $m = n$. Then we immediately obtain $\exists P' \in \Pi.\ P \equiv (\nu n)P' \wedge P' \vDash \mathcal{A}$.

Suppose $m \neq n$. Since $n \notin fn(P)$ by assumption and $P \equiv (\nu m)P'$ we have $n \notin fn(P')$ by Lemma 2-1(1). Take $P'' = P'\{m \leftarrow n\}$. By Lemma 3-3, since $n \notin fn(P', \mathcal{A}\{n \leftarrow m\})$ and $P' \vDash \mathcal{A}\{n \leftarrow m\}$, we obtain $P'\{m \leftarrow n\} \vDash \mathcal{A}\{n \leftarrow m\}\{m \leftarrow n\}$, that is, $P'' \vDash \mathcal{A}$. Now, $P \equiv (\nu m)P' = (\nu n)P'\{m \leftarrow n\} = (\nu n)P''$, so that $P \equiv (\nu n)P''$. We have shown that $\exists P'' \in \Pi.\ P \equiv (\nu n)P'' \wedge P'' \vDash \mathcal{A}$.

Case $\Leftarrow$)

Assume $\exists P' \in \Pi.\ P \equiv (\nu n)P' \wedge P' \vDash \mathcal{A}$; then $n \notin fn(P)$ by Lemma 2-1(1). Moreover:

$\exists P' \in \Pi.\ P \equiv (\nu n)P' \wedge P' \vDash \mathcal{A} \Rightarrow P \vDash n \circledR \mathcal{A}$

$\Rightarrow P \vDash (x \circledR \mathcal{A}\{n \leftarrow x\})\{x \leftarrow n\}$ (since $n \circledR \mathcal{A}$ is closed and hence $x \notin fv(\mathcal{A})$)

$\Rightarrow \exists n \in \Lambda.\ n \notin fn(P, x \circledR (\mathcal{A}\{n \leftarrow x\})) \wedge P \vDash (x \circledR \mathcal{A}\{n \leftarrow x\})\{x \leftarrow n\}$

$\Rightarrow P \vDash \mathcal{N}x.x \circledR \mathcal{A}\{n \leftarrow x\} \Rightarrow P \vDash \mathsf{H}x.(\mathcal{A}\{n \leftarrow x\})$

□

Remark. We saw that for the existential definition of the hidden-name quantifier, we obtained the undesirable property $(\nu n)n[] \vDash \exists x.x \circledR p[]$. Since $n[] \nvDash p[]$, Proposition 7-3 implies that $(\nu n)n[] \nvDash \mathcal{N}x.x \circledR p[]$. As a sanity check, suppose that $(\nu n)n[] \vDash \mathcal{N}x.x \circledR p[]$; this would mean that $\exists m \in \Lambda.\ m \notin fn((\nu n)n[], p[]) \wedge \exists P' \in \Pi.\ (\nu n)n[] \equiv (\nu m)P' \wedge P' \vDash p[]$. But $P' \vDash p[]$ implies $P' \equiv p[]$, and since $m \neq p$ we cannot have $(\nu n)n[] \equiv (\nu m)P'$ by Lemma 2-1(1). □

Remark. We saw that for the universal definition of the hidden-name quantifier, we obtained the undesirable property $(\nu n)p[] \nvDash \forall x.x \circledR \neg x[]$. Since $p[] \vDash \neg n[]$, Proposition 7-3 implies that $(\nu n)p[] \vDash \mathcal{N}x.x \circledR \neg x[]$. In fact, $p[] \vDash \neg n[]$ implies $(\nu n)p[] \vDash n \circledR \neg n[]$, which is the same as $(\nu n)p[] \vDash (x \circledR \neg x[])\{x \leftarrow n\}$, where $n \notin fn((\nu n)p[], x \circledR \neg x[])$. Therefore, $\exists m \in \Lambda.\ m \notin fn((\nu n)p[], x \circledR \neg x[]) \wedge (\nu n)p[] \vDash (x \circledR \neg x[])\{x \leftarrow m\}$, which means $(\nu n)p[] \vDash \mathcal{N}x.x \circledR \neg x[]$ by Lemma 6-5. □

Remark. We obtain $\forall x.\ x \circledR \mathcal{A} \vdash \mathsf{H}x.\mathcal{A} \vdash \exists x.\ x \circledR \mathcal{A}$. However, there are no interesting rules for $\neg \mathsf{H}x.\mathcal{A}$. □

Remark. This fails:

$$\left.\begin{array}{l}\\ \end{array}\right\} \mathsf{H}x.\mathcal{A} \vdash \mathcal{A} \qquad \text{where } x \notin fv(\mathcal{A})$$

because $(\nu n)(n[] \mid n[]) \vDash \mathcal{N}x.x \circledR (\neg \mathbf{0} \mid \neg \mathbf{0})$ but $(\nu n)(n[] \mid n[]) \nvDash \neg \mathbf{0} \mid \neg \mathbf{0}$. This is ®'s fault, not $\mathcal{N}$'s: $n \circledR \mathcal{A} \vdash \mathcal{A}$ fails with the same counterexample. □

Remark.

Property (νx-proper), Proposition 7-3, is derivable within the logic. That is:

$$n \circledR \mathbf{T} \wedge \mathsf{H}x.(\mathcal{A}\{n \leftarrow x\}) \dashv\vdash n \circledR \mathcal{A} \qquad \text{where } x \notin fv(\mathcal{A}):$$

In the left-to-right direction, from $n \circledR \mathbf{T} \wedge \mathsf{H}x.(\mathcal{A}\{n \leftarrow x\})$ we obtain by definition $n \circledR \mathbf{T} \wedge \mathcal{N}x.x \circledR \mathcal{A}\{n \leftarrow x\}$, and by ($\mathcal{N}$ $\forall$) we obtain $n \circledR \mathbf{T} \wedge \forall x.\ x \# fnv(x \circledR \mathcal{A}\{n \leftarrow x\}) \text{-} \{x\} \wedge x \circledR \mathbf{T} \Rightarrow x \circledR \mathcal{A}\{n \leftarrow x\}$, which since $x \notin fv(\mathcal{A})$ is the same as $n \circledR \mathbf{T} \wedge \forall x.\ x \# fnv(\mathcal{A}) \text{-} \{n\} \wedge x \circledR \mathbf{T} \Rightarrow x \circledR \mathcal{A}\{n \leftarrow x\}$. By instantiating x to n, we get $n \circledR \mathbf{T} \wedge (n \# fnv(\mathcal{A}) \text{-} \{n\} \wedge n \circledR \mathbf{T} \Rightarrow n \circledR \mathcal{A}\{n \leftarrow n\})$. Since $n \# fnv(\mathcal{A}) \text{-} \{n\}$ is true and is $n \circledR \mathbf{T}$ assumed, we obtain $n \circledR \mathcal{A}\{n \leftarrow n\}$, that is $n \circledR \mathcal{A}$.

In the right-to-left direction, assuming $n \circledR \mathcal{A}$, from $\mathcal{A} \vdash \mathbf{T}$ and ($\circledR \vdash$) we obtain $n \circledR \mathcal{A} \vdash n \circledR \mathbf{T}$ and hence the first conjunct, $n \circledR \mathbf{T}$. Then we trivially obtain $n \# fnv(n \circledR \mathcal{A}\{n \leftarrow n\})\text{-}\{n\} \wedge n \circledR \mathbf{T} \wedge n \circledR \mathcal{A}\{n \leftarrow n\}$, since these conjuncts are either true or implied by $n \circledR \mathcal{A}$. That formula is the same as $(x \# fnv(x \circledR \mathcal{A}\{n \leftarrow x\})\text{-}\{x\} \wedge x \circledR \mathbf{T} \wedge x \circledR \mathcal{A}\{n \leftarrow x\})\{x \leftarrow n\}$ since $x \notin fv(\mathcal{A})$. Then, by ($\exists$ R) we obtain $\exists x.\ x \# fnv(x \circledR \mathcal{A}\{n \leftarrow x\})\text{-}\{x\} \wedge x \circledR \mathbf{T} \wedge x \circledR \mathcal{A}\{n \leftarrow x\}$. This implies, by ($\forall\!\!\!\forall\ \exists$), that $\forall\!\!\!\forall x. x \circledR \mathcal{A}\{n \leftarrow x\}$, which is the same as $\mathsf{H}x.(\mathcal{A}\{n \leftarrow x\})$. □

7.3 Example

As an example of a specification containing a hidden-name quantifier, consider a situation where a secret is shared by two locations n and m, but is not known outside those locations.

We can state this as follows (recall that $\copyright\eta \triangleq \neg\eta \circledR \mathbf{T}$ and that $P \models \copyright n$ iff $n \in fn(P)$):

$$\mathsf{H}x.\ (n[\copyright x] \mid m[\copyright x])$$

It reads: for a fresh x, the name x is known at n and m, and is restricted anywhere else.

Expanding the definitions, we obtain:

$P \models \mathsf{H}x.\ (n[\copyright x] \mid m[\copyright x])$

$\Leftrightarrow P \models \forall\!\!\!\forall x. x \circledR (n[\copyright x] \mid m[\copyright x])$

$\Leftrightarrow \exists r \in \Lambda.\ r \notin fn(P) \cup \{n,m\} \wedge \exists Q \in \Pi.\ P \equiv (\nu r)Q \wedge Q \models (n[\copyright x] \mid m[\copyright x])\{x \leftarrow r\}$

$\Leftrightarrow \exists r \in \Lambda.\ r \notin fn(P) \cup \{n,m\} \wedge \exists Q \in \Pi.\ P \equiv (\nu r)Q \wedge Q \models n[\copyright r] \mid m[\copyright r]$

$\Leftrightarrow \exists r \in \Lambda.\ r \notin fn(P) \cup \{n,m\} \wedge \exists Q \in \Pi.\ P \equiv (\nu r)Q \wedge$
$\quad \exists Q', Q'' \in \Pi.\ Q \equiv Q' \mid Q'' \wedge Q' \models n[\copyright r] \wedge Q'' \models m[\copyright r]$

$\Leftrightarrow \exists r \in \Lambda.\ r \notin fn(P) \cup \{n,m\} \wedge \exists Q \in \Pi.\ P \equiv (\nu r)Q \wedge$
$\quad \exists R', R'' \in \Pi.\ Q \equiv n[R'] \mid m[R''] \wedge R' \models \copyright r \wedge R'' \models \copyright r$

$\Leftrightarrow \exists r \in \Lambda.\ r \notin fn(P) \cup \{n,m\} \wedge \exists R', R'' \in \Pi.\ P \equiv (\nu r)(n[R'] \mid m[R'']) \wedge r \in fn(R') \wedge r \in fn(R'')$

The last line reads: P satisfies the specification iff there exists a name r that is fresh (not conflicting with n and m or public to P), such that r is known to the processes R' and R'' located at n and m, and is restricted inside P.

Here is a simple example of an implementation of this specification:

$$P = (\nu p)\ (n[p[\,]] \mid m[p[\,]])$$

In process calculi, it is common to extrude the scope of a hidden name, to prepare for the interaction between processes within a restricted scope and processes outside the scope. We can do something similar at the logical level, extruding the scope of hiding quantifiers. For example, we can infer, logically, that:

$$(\mathsf{H}x.\ \mathcal{A}) \mid \mathcal{B} \vdash \mathsf{H}x.\ (\mathcal{A} \mid x \circledR \mathcal{B})$$

That is, suppose we start with a specification of the form $\mathsf{H}x.\ \mathcal{A}$ and we composed it with another specification to obtain $(\mathsf{H}x.\ \mathcal{A}) \mid \mathcal{B}$, we may need to extrude $\mathsf{H}x$ around $\mathcal{B}$ (assuming *w.l.o.g.* that x is not free in $\mathcal{B}$). We first use the property $\mathcal{B} \vdash \mathsf{H}x.\ \mathcal{B}$ for x not free in $\mathcal{B}$ (Logical Corollary (H *fv*) in 10-14) to infer $(\mathsf{H}x.\ \mathcal{A}) \mid (\mathsf{H}x.\ \mathcal{B})$. Then we use the property $(\mathsf{H}x.\ \mathcal{A}) \mid (\mathsf{H}x.\ \mathcal{B}) \vdash \mathsf{H}x.\ (\mathcal{A} \mid x \circledR \mathcal{B})$ (Logical Corollary ($\mathsf{H} \mid$) in 10-14). Thus, we infer the

specification $\mathsf{H}x.\ (\mathcal{A} \mid x\circledR\mathcal{B})$, where $\mathsf{H}x$ has been extruded, and where $x\circledR$ guarantees that the choice of x does not clash with any name in the process that originally independently satisfies $\mathcal{B}$.

8 Connections with Other Logics

8.1 Relevant Logic

The shape of the definition of the satisfaction relation turns out to be very similar to Urquhart's semantics of relevant logic [34]. (Thanks to Peter O'Hearn and David Pym for pointing this out.) In particular $\mathcal{A}|\mathcal{B}$ is similar to *intensional conjunction*, and $\mathcal{A}\triangleright\mathcal{B}$ is similar to *relevant implication* in that semantics.

The main difference is that we do not have contraction: this rule is not sound for process calculi, because $P|P \neq P$ under any reasonable equivalence. Urquhart's semantics without contraction, or even relevant logics without contraction, do not seem to have been considered.

Moreover, we use an equivalence, $\equiv$, instead of a Kripke-style partial order as in Urquhart's general case. If we were to adopt a partial order (perhaps some asymmetric form of structural congruence), then the classical fragment of our logic would have to be replaced by an intuitionistic fragment, in order to maintain the analogue of Lemma 3-1. This seems to be the deep reason why we can get by with classical implication.

8.2 Bunched Logic

Peter O'Hearn and David Pym study *bunched logics* [31], where sequents have two structural combinators, instead of the standard single "," combinator (usually meaning $\wedge$ or $\otimes$ on the left) found in most presentations of logic. Thus, sequents are *bunches* of formulas, instead of lists of formulas. Correspondingly, there are two implications that arise as the adjuncts of the two structural combinators.

The situation is very similar to our combinators $|$ and $\wedge$, which can combine to irreducible bunches of formulas in sequents, and to our two implications $\Rightarrow$ and $\triangleright$. However, we have a classical and a linear implication, while bunched logics have so far had an intuitionistic and a linear implication.

8.3 Linear Logic

We now relate a fragment of our logic to intuitionistic linear logic. Although the connections with some parts of linear logic are slightly degenerate, we can make them quite precise. We omit the proofs; these are not hard, except that deriving the rules of linear logic within our logic requires some experience.

First note that, when considering $|$ as a structural connective, we must reject weakening, which entails $\mathcal{A} \vdash \mathbf{0}$, and contraction, which entails $\mathcal{A} \vdash \mathcal{A} \mid \mathcal{A}$: both are unsound in our process model. Therefore, we are at least somewhat close in spirit to linear logic. Our sequents are linear in the sense that we must have the same number of process components

on the left and right of $\vdash$. In other words, space cannot be instantaneously created or destroyed. Consequently, the implication $\triangleright$ arising as an adjunct of $|$ is a linear implication: note that in $\mathcal{A} \triangleright \mathcal{B}$ the attacker that satisfies $\mathcal{A}$ is used exactly once in the system the satisfies $\mathcal{B}$.

Multiplicative intuitionistic linear logic (MILL) can be captured faithfully by identifying $\multimap_{\text{MILL}} = \triangleright$, $\otimes_{\text{MILL}} = |$, and $\mathbf{1}_{\text{MILL}} = \mathbf{0}$: the rules of MILL and the subset of our rules that involve only those connectives (plus a derivable cut rule for $|$ corresponding to the MILL cut rule) are interderivable. However, this precise match is obtained by paring down both linear logic and our logic. We can go further and draw a connection with full intuitionistic linear logic (ILL [21,24,25,28]), both syntactically and semantically. The discrepancies with ILL are as follows. We identify $\bot_{\text{ILL}}$ and $\mathbf{0}_{\text{ILL}}$ (as $\mathbf{F}$); therefore, $\mathcal{A}^{\bot}$ acquires special properties. The additives $\oplus$ and & distribute over each other (both semantically and as a derived rule). The semantic interpretation of $!\mathcal{A}$ is rather degenerate; in particular, $!\mathcal{A} \multimap \mathcal{B}$ does not seem to have an interesting interpretation. The inference rules of ILL are listed in the proof of Proposition 8-9.

Quantales

A Quantale [21] is a complete join semilattice with a commutative monoid structure that distributes over join. That is:

8-1 Definition (Quantale)

A quantale $\mathcal{Q}$ is a structure $\langle S : \text{Set}, \leq : S^2 \to \text{Bool}, \bigvee : \mathcal{P}(S) \to S, \otimes : S^2 \to S, 1 : S \rangle$ such that for any $p,q,r \in S$ and $Q \subseteq S$:

$p \leq p$

$p \leq q \ \wedge \ q \leq r \ \Rightarrow \ p \leq r$

$p \leq q \ \wedge \ q \leq p \ \Rightarrow \ p = q$

$p \otimes (q \otimes r) \ = \ (p \otimes q) \otimes r$

$p \otimes q \ = \ q \otimes p$

$p \otimes 1 \ = \ p$

$\bigvee Q \in S$

$\forall q \in Q.\, q \leq \bigvee Q$

$(\forall q \in Q.\, q \leq p) \ \Rightarrow \ \bigvee Q \leq p$

$p \otimes \bigvee Q \ = \ \bigvee \{ p \otimes q \parallel q \in Q \}$

□

Quantales are models of intuitionistic linear logic, according to the following mapping $[\![\mathcal{A}]\!]_{\mathcal{Q}}$ (we omit the subscript when $\mathcal{Q}$ is unambiguous), where $A \vee B \triangleq \bigvee \{A, B\}$, $A \wedge B \triangleq \bigvee \{ C \parallel C \leq A \wedge C \leq B \}$, and $\upsilon X.\, A\{X\} \triangleq \bigvee \{ C \parallel C \leq A\{C\} \}$.

8-2 Definition (ILL)

(1) $[\![\mathbf{1}_{\text{ILL}}]\!] \ \triangleq \ 1$

$[\![\bot_{\text{ILL}}]\!] \ \triangleq$ any element of S

$[\![\top_{\text{ILL}}]\!] \ \triangleq \ \bigvee S$

$[\![\mathbf{0}_{\text{ILL}}]\!] \ \triangleq \ \bigvee \emptyset$

$[\![\mathcal{A} \oplus \mathcal{B}]\!] \triangleq [\![\mathcal{A}]\!] \vee [\![\mathcal{B}]\!]$

$[\![\mathcal{A} \,\&\, \mathcal{B}]\!] \triangleq [\![\mathcal{A}]\!] \wedge [\![\mathcal{B}]\!]$

$[\![\mathcal{A} \otimes \mathcal{B}]\!] \triangleq [\![\mathcal{A}]\!] \otimes [\![\mathcal{B}]\!]$

$[\![\mathcal{A} \multimap \mathcal{B}]\!] \triangleq \bigvee\{C \parallel C \otimes [\![\mathcal{A}]\!] \leq [\![\mathcal{B}]\!]\}$

$[\![!\mathcal{A}]\!] \triangleq \upsilon X.\ 1 \,\&\, [\![\mathcal{A}]\!] \,\&\, (X \otimes X)$

(2) $\boldsymbol{vld}_{\mathrm{ILL}}(\mathcal{A}_1, ..., \mathcal{A}_n \vdash_{\mathrm{ILL}} \mathcal{B})_{\mathcal{Q}} \triangleq [\![\mathcal{A}_1]\!]_{\mathcal{Q}} \otimes_{\mathcal{Q}} ... \otimes_{\mathcal{Q}} [\![\mathcal{A}_n]\!]_{\mathcal{Q}} \leq_{\mathcal{Q}} [\![\mathcal{B}]\!]_{\mathcal{Q}}$

In particular, $\boldsymbol{vld}_{\mathrm{ILL}}(\vdash_{\mathrm{ILL}} \mathcal{B})_{\mathcal{Q}}$ iff $1_{\mathcal{Q}} \leq_{\mathcal{Q}} [\![\mathcal{B}]\!]_{\mathcal{Q}}$.

(3) $\boldsymbol{vld}_{\mathrm{ILL}}(\mathcal{A}_1, ..., \mathcal{A}_n \vdash_{\mathrm{ILL}} \mathcal{B}) \triangleq$ for all quantales $\mathcal{Q}$, $\boldsymbol{vld}_{\mathrm{ILL}}(\mathcal{A}_1, ..., \mathcal{A}_n \vdash_{\mathrm{ILL}} \mathcal{B})_{\mathcal{Q}}$

□

The following theorem is folklore:

8-3 Proposition (Soundness and Completeness)

$\mathcal{A}_1, ..., \mathcal{A}_n \vdash_{\mathrm{ILL}} \mathcal{B}$ iff $\boldsymbol{vld}_{\mathrm{ILL}}(\mathcal{A}_1, ..., \mathcal{A}_n \vdash_{\mathrm{ILL}} \mathcal{B})$.

□

The Process Quantale

8-4 Proposition (Process Quantale)

The structure $\Theta \triangleq \langle \Phi, \subseteq, \bigcup, \otimes, \mathbf{1} \rangle$ is a quantale, where, for $A,B \subseteq \Pi$:

$A^{\equiv} \triangleq \{P \parallel \exists Q \in A.\ P \equiv Q\}$

$\Phi \triangleq \{A^{\equiv} \parallel A \subseteq \Pi\}$

$A \otimes B \triangleq \{P|Q \parallel P \in A \wedge Q \in B\}^{\equiv}$

$1 \triangleq \{\mathbf{0}\}^{\equiv}$

Proof

$\otimes : \Phi^2 \to \Phi$, by definition.

$1 : \Phi$, by definition.

$\bigcup : \mathcal{P}(\Phi) \to \Phi$, since a union of $\equiv$-closed sets is $\equiv$-closed.

$A \otimes (B \otimes C) = \{P|S \parallel P \in A \wedge S \in B \otimes C\}^{\equiv}$

$= \{P|S \parallel P \in A \wedge S \in \{Q|R \parallel Q \in B \wedge R \in C\}^{\equiv}\}^{\equiv}$

$= \{T \parallel T \equiv P|S \wedge P \in A \wedge S \equiv Q|R \wedge Q \in B \wedge R \in C\}$

$= \{T \parallel T \equiv P|(Q|R) \wedge P \in A \wedge Q \in B \wedge R \in C\}$

$= \{T \parallel T \equiv (P|Q)|R \wedge P \in A \wedge Q \in B \wedge R \in C\}$

$= ... = (A \otimes B) \otimes C$

$A \otimes B = \{P|Q \parallel P \in A \wedge Q \in B\}^{\equiv}$

$= \{Q|P \parallel P \in A \wedge Q \in B\}^{\equiv}$

$= B \otimes A$

$A \otimes \{\mathbf{0}\}^{\equiv} = \{P|Q \parallel P \in A \wedge Q \in \{\mathbf{0}\}^{\equiv}\}^{\equiv}$

$= \{T \parallel T \equiv P|Q \wedge P \in A \wedge Q \equiv S \wedge S = \mathbf{0}\}$

$= \{T \parallel T \equiv P|\mathbf{0} \wedge P \in A\}$

$= \{T \parallel T \equiv P \wedge P \in A\}$

$= A$

$A \otimes \bigcup\Theta = \{P|Q \parallel P \in A \wedge Q \in \bigcup\Theta\}^{\equiv}$

$= \{T \parallel T \equiv P|Q \wedge P \in A \wedge Q \in \bigcup\Theta\}$

$= \bigcup\{\{T \parallel T \equiv P|Q \wedge P \in A \wedge Q \in B\} \parallel B \in \Theta\}$

$= \bigcup\{\{P|Q \parallel P \in A \wedge Q \in B\}^{\equiv} \parallel B \in \Theta\}$

$= \bigcup\{A \otimes B \parallel B \in \Theta\}$

□

Our logic is interpreted as follows:

8-5 Definition (Interpretation in Θ)

$[\![\mathcal{A}]\!] \triangleq \{P{:}\Pi \parallel P \vDash \mathcal{A}\}$

□

Note that, by Proposition 3-1, $[\![\mathcal{A}]\!] = [\![\mathcal{A}]\!]^{\equiv}$.

Remark. It would be possible to consider "enriched quantales" that have additional structure corresponding to ambient operations. That is, we could add to the structure an operator -[-]: $\Lambda \times S \to S$ that distributes over join: $n[\bigvee Q] = \bigvee\{n[q] \parallel q \in Q\}$. (In the process quantale, we would have $n[A] \triangleq \{n[P] \parallel P \in A\}^{\equiv}$.) We could then add to ILL the connectives $n[\mathcal{A}]$ and $\mathcal{A}@n$, with interpretation:

$[\![n[\mathcal{A}]]\!] \triangleq n[[\![\mathcal{A}]\!]]$

$[\![\mathcal{A}@n]\!] \triangleq \bigvee\{C \parallel n[C] \leq [\![\mathcal{A}]\!]\}$

A similar enrichment could be obtained also for $x \circledR \mathcal{A}$. However, we do not pursue this further.

□

Embedding Intuitionistic Linear Logic

In this paper, the linear constants $\mathbf{1}, \top, \mathbf{0}$ are known respectively as $\mathbf{0}, \mathbf{T}, \mathbf{F}$, and are interpreted in the process quantale as follows: $\mathbf{0} = \{\mathbf{0}\}^{\equiv}$, $\mathbf{T} = \Pi$, $\mathbf{F} = \emptyset$.

As for the fourth linear constant, $\bot$, there are a few possible choices in Φ, producing alternative linear logics:

- $\bot = \emptyset$. Then $\bot$ has the sense of unsatisfiability. We have the identification $\bot = \mathbf{F}$, and $\mathcal{A}^{\bot}$ is what we have called $\mathcal{A}^{\mathbf{F}}$, with its additional axioms. This seems to us the only really reasonable choice.
- $\bot = \{\mathbf{0}\}^{\equiv}$. Then $\bot$ has the sense of (trivially) deadlocked processes, which might appear more in tune with an interpretation of linear logic as a theory of concurrency. We have the identification, $\bot = \mathbf{0}$, and $\mathcal{A}^{\bot}$ is what we might call $\mathcal{A}^{\mathbf{0}}$. However, the notion of $\mathcal{A}^{\mathbf{0}}$, the set of processes that when composed with $\mathcal{A}$ processes produce $\mathbf{0}$, seems pretty trivial in most process calculi.
- Factoring over a more general equivalence ≈, rather than simply ≡, would give us a better notion of deadlocked processes. We could perhaps keep $\mathbf{0} = \{\mathbf{0}\}^{\equiv}$ and take $\bot$

$= \{\mathbf{0}\}^{\approx}$. Still, under most equivalences, there are not many processes that can neutralize other processes by simple composition.

Therefore, we embed intuitionistic linear logic into our logic as follows:

8-6 Definition (Embedding ILL)

$\mathbf{1}_{\mathrm{ILL}} \triangleq \mathbf{0}$
$\bot_{\mathrm{ILL}} \triangleq \mathbf{F}$
$\top_{\mathrm{ILL}} \triangleq \mathbf{T}$
$\mathbf{0}_{\mathrm{ILL}} \triangleq \mathbf{F}$
$\mathcal{A} \oplus \mathcal{B} \triangleq \mathcal{A} \vee \mathcal{B}$
$\mathcal{A} \,\&\, \mathcal{B} \triangleq \mathcal{A} \wedge \mathcal{B}$
$\mathcal{A} \otimes \mathcal{B} \triangleq \mathcal{A} \mid \mathcal{B}$
$\mathcal{A} \multimap \mathcal{B} \triangleq \mathcal{A} \rhd \mathcal{B}$
$!\mathcal{A} \triangleq \mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}}$

□

For these definitions to be meaningful, we need to show that the logical constants and operators defined on the left match the corresponding semantic constants and operators in the quantale Θ, according to Definition 8-2.

8-7 Proposition (Soundness of the ILL interpretation)

The linear constants and operators correspond to their quantale definitions in Θ.

Proof

We show only the most interesting cases:

$\otimes$) Show $[\![\mathcal{A} \otimes \mathcal{B}]\!] = [\![\mathcal{A}]\!] \otimes [\![\mathcal{B}]\!]$.

$P \in [\![\mathcal{A} \otimes \mathcal{B}]\!] \;\Leftrightarrow\; P \in [\![\mathcal{A} \mid \mathcal{B}]\!] \;\Leftrightarrow\; P \models \mathcal{A} \mid \mathcal{B}$

$\Leftrightarrow\; \exists P', P'' : \Pi.\; P \equiv P' \mid P'' \wedge P' \models \mathcal{A} \wedge P'' \models \mathcal{B}$

$\Leftrightarrow\; P \in \{P' \mid P'' \parallel P' \models \mathcal{A} \wedge P'' \models \mathcal{B}\}^{\equiv}$

$\Leftrightarrow\; P \in \{P' \mid P'' \parallel P' \in [\![\mathcal{A}]\!] \wedge P'' \in [\![\mathcal{B}]\!]\}^{\equiv}$

$\Leftrightarrow\; P \in [\![\mathcal{A}]\!] \otimes [\![\mathcal{B}]\!]$

$\multimap$) Show $[\![\mathcal{A} \multimap \mathcal{B}]\!] = [\![\mathcal{A}]\!] \multimap [\![\mathcal{B}]\!]$.

Let $A = [\![\mathcal{A}]\!]$ and $B = [\![\mathcal{B}]\!]$.

$P \in [\![\mathcal{A}]\!] \multimap [\![\mathcal{B}]\!] \;\Leftrightarrow\; P \in A \multimap B \;\Leftrightarrow\; P \in \bigcup\{C \parallel C \otimes A \subseteq B\}$

$\Leftrightarrow \exists C.\; P \in C \wedge C \otimes A \subseteq B$

$\Leftrightarrow \exists C.\; P \in C \wedge \forall Q.\; (\exists Q', Q''.\; Q \equiv Q' \mid Q'' \wedge Q' \in C \wedge Q'' \in A) \Rightarrow Q \in B$

$\Leftrightarrow \forall Q''.\; Q'' \in A \Rightarrow P \mid Q'' \in B$

The last step works as follows:

1) Assume $\exists C.\; P \in C \wedge \forall Q.\; (\exists Q', Q''.\; Q \equiv Q' \mid Q'' \wedge Q' \in C \wedge Q'' \in A) \Rightarrow Q \in B$. Take any R and assume $R \in A$. Instantiate the assumption with $P \mid R$ for Q and take $Q' = P$ and $Q'' = R$; we obtain $P \mid R \in B$.

2) Conversely, assume $\forall R.\ R \in A \Rightarrow P|R \in B$. Take $C=\{P\}^{\equiv}$, take any Q, and assume $(\exists Q', Q''.\ Q \equiv Q'|Q'' \wedge Q' \in \{P\}^{\equiv} \wedge Q'' \in A)$. Instantiating the assumption with Q'' for R, we obtain $P|Q'' \in B$. Now, $Q' \equiv P$ by assumption, hence $P|Q'' \equiv Q'|Q'' \equiv Q$. Since B is $\equiv$-closed, we obtain $Q \in B$.

Hence $P \in [\![\mathcal{A}]\!] \multimap [\![\mathcal{B}]\!] \Leftrightarrow \forall Q''.\ Q'' \in A \Rightarrow P|Q'' \in B \Leftrightarrow \forall Q''.\ Q'' \vDash \mathcal{A} \Rightarrow P|Q'' \vDash \mathcal{B}$ $\Leftrightarrow P \vDash \mathcal{A} \triangleright \mathcal{B} \Leftrightarrow P \in [\![\mathcal{A} \triangleright \mathcal{B}]\!] \Leftrightarrow P \in [\![\mathcal{A} \multimap \mathcal{B}]\!]$.

!) Show $[\![!\mathcal{A}]\!] = !\,[\![\mathcal{A}]\!]$.

First we show that $\forall P.\ \mathbf{0} \vDash \mathcal{A} \Leftrightarrow P \vDash (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}}$.

Take any P; by definition of $\triangleright$, we have $P \vDash (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}} \Leftrightarrow (\forall Q.\ Q \vDash \mathbf{0} \Rightarrow \mathcal{A})$. Then, $(\forall Q.\ Q \vDash \mathbf{0} \Rightarrow \mathcal{A}) \Leftrightarrow (\forall Q.\ Q \vDash \mathbf{0} \Rightarrow Q \vDash \mathcal{A}) \Leftrightarrow \mathbf{0} \vDash \mathcal{A}$. The last step is by instantiation of Q with $\mathbf{0}$, in one direction, and by Proposition 3-1, in the other direction.

Then we compute: $P \in [\![!\mathcal{A}]\!] \Leftrightarrow P \in [\![\mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}}]\!] \Leftrightarrow P \in [\![\mathbf{0}]\!] \wedge P \in [\![(\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}}]\!]$ $\Leftrightarrow P \equiv \mathbf{0} \wedge P \vDash (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}} \Leftrightarrow P \equiv \mathbf{0} \wedge \mathbf{0} \vDash \mathcal{A}$.

Now, in a quantale $!A = \upsilon X.\ 1\ \&\ A\ \&\ (X \otimes X)$, which in Θ means $\upsilon X.\ \{\mathbf{0}\}^{\equiv} \cap A \cap (X \mid X)$. If $\mathbf{0} \notin A$ then $\{\mathbf{0}\}^{\equiv} \cap A = \emptyset$, and $!A = \emptyset$. If instead $\mathbf{0} \in A$, then $\{\mathbf{0}\}^{\equiv} \cap A = \{\mathbf{0}\}^{\equiv}$, and $!A = \upsilon X.\ \{\mathbf{0}\}^{\equiv} \cap (X \mid X)$. We have that $\{\mathbf{0}\}^{\equiv}$ is a fixpoint of $\lambda X.\ \{\mathbf{0}\}^{\equiv} \cap (X \mid X)$; moreover, if $B = \{\mathbf{0}\}^{\equiv} \cap (B \mid B)$ then $B \subseteq \{\mathbf{0}\}^{\equiv}$, hence $\{\mathbf{0}\}^{\equiv}$ is the greatest fixpoint, and $!A = \{\mathbf{0}\}^{\equiv}$. In conclusion: if $\mathbf{0} \notin A$ then $!A = \emptyset$ else if $\mathbf{0} \in A$ then $!A = \{\mathbf{0}\}^{\equiv}$ and, by contrapositive, if $!A \neq \emptyset$ then $\mathbf{0} \in A$.

Hence $P \in !\,[\![\mathcal{A}]\!] \Rightarrow !\,[\![\mathcal{A}]\!] \neq \emptyset \Rightarrow \mathbf{0} \in [\![\mathcal{A}]\!] \Rightarrow !\,[\![\mathcal{A}]\!] = \{\mathbf{0}\}^{\equiv} \Rightarrow P \in \{\mathbf{0}\}^{\equiv}$; that is $P \in !\,[\![\mathcal{A}]\!] \Rightarrow P \equiv \mathbf{0} \wedge \mathbf{0} \vDash \mathcal{A}$. Conversely, if $P \equiv \mathbf{0} \wedge \mathbf{0} \vDash \mathcal{A}$, then $\mathbf{0} \in [\![\mathcal{A}]\!] \Rightarrow !\,[\![\mathcal{A}]\!] = \{\mathbf{0}\}^{\equiv} \Rightarrow P \in !\,[\![\mathcal{A}]\!]$.

In conclusion $P \in !\,[\![\mathcal{A}]\!] \Leftrightarrow P \equiv \mathbf{0} \wedge \mathbf{0} \vDash \mathcal{A} \Leftrightarrow P \in [\![!\mathcal{A}]\!]$.

□

Moreover, in our model the linear notion of validity matches our notion of validity:

8-8 Proposition

Let $\mathcal{A}_1, ..., \mathcal{A}_n, \mathcal{B}$ be formulas in ILL.
$\boldsymbol{vld}_{\mathrm{ILL}}(\mathcal{A}_1, ..., \mathcal{A}_n \vdash_{\mathrm{ILL}} \mathcal{B})_{\Theta} \Leftrightarrow \boldsymbol{vld}(\mathcal{A}_1 \mid ... \mid \mathcal{A}_n \vdash \mathcal{B})$
(For n=0 this means: $\boldsymbol{vld}_{\mathrm{ILL}}(\vdash_{\mathrm{ILL}} \mathcal{B})_{\Theta} \Leftrightarrow \boldsymbol{vld}(\mathbf{0} \vdash \mathcal{B})$.)

Proof

$(\forall P.\ P \vDash \mathcal{A}_1|...|\mathcal{A}_n \Rightarrow \mathcal{B}) \Leftrightarrow (\forall P.\ P \vDash \mathcal{A}_1|...|\mathcal{A}_n \Rightarrow P \vDash \mathcal{B})$
$\Leftrightarrow (\forall P.\ P \in [\![\mathcal{A}_1|...|\mathcal{A}_n]\!] \Rightarrow P \in [\![\mathcal{B}]\!]) \Leftrightarrow [\![\mathcal{A}_1|...|\mathcal{A}_n]\!] \subseteq [\![\mathcal{B}]\!]$
$\Leftrightarrow [\![\mathcal{A}_1]\!] \otimes ... \otimes [\![\mathcal{A}_n]\!] \subseteq [\![\mathcal{B}]\!]$

□

Furthermore, we can derive the rules of intuitionistic linear logic within our logic. In particular, we can derive the "strong" rules for $!\mathcal{A}$ that correspond to an interpretation of **!** as a maximal fixpoint [24, 28, 21].

8-9 Proposition

Derivations in ILL can be mapped to derivations in the ambient logic. More precisely, let $\mathcal{A}_1, ..., \mathcal{A}_n, \mathcal{B}$ be formulas in ILL, then:
$\mathcal{A}_1, ..., \mathcal{A}_n \vdash_{\mathrm{ILL}} \mathcal{B} \;\Rightarrow\; (...(\mathcal{A}_1) \mid ...) \mid \mathcal{A}_n \vdash \mathcal{B}$
(Where for n=0 this means: $\vdash_{\mathrm{ILL}} \mathcal{B} \;\Rightarrow\; \mathbf{0} \vdash \mathcal{B}$.)

Proof

We show that the inference rules of ILL [21] are derivable in the ambient logic, under the operator mapping of Definition 8-6 and the sequent mapping shown in the statement above. (We often omit the steps involving just the reassociation of $(...(\mathcal{A}_1) \mid ...) \mid \mathcal{A}_n$ on the left of $\vdash$.) We use nested brackets $\lceil ... \rceil$ to indicate the structure of proof trees.

(Id) $\vdash_{\mathrm{ILL}} \mathcal{A} \vdash_{\mathrm{ILL}} \mathcal{A}$

We show: $\vdash \mathcal{A} \vdash \mathcal{A}$. Simply by (Id).

(Cut) $\mathcal{F} \vdash_{\mathrm{ILL}} \mathcal{A};\ \mathcal{G}, \mathcal{A} \vdash_{\mathrm{ILL}} \mathcal{B} \vdash_{\mathrm{ILL}} \mathcal{F}, \mathcal{G} \vdash_{\mathrm{ILL}} \mathcal{B}$

We show: $\mathcal{F} \vdash \mathcal{A};\ \mathcal{G} \mid \mathcal{A} \vdash \mathcal{B} \vdash \mathcal{F} \mid \mathcal{G} \vdash \mathcal{B}$.

$\lceil \lceil \mathcal{F} \vdash \mathcal{A};\ \lceil \mathcal{G} \mid \mathcal{A} \vdash \mathcal{B} \vdash_{(\mid \triangleright)} \mathcal{G} \vdash \mathcal{A} \triangleright \mathcal{B} \rceil \rceil \vdash_{(\mid \vdash)} \mathcal{F} \mid \mathcal{G} \vdash \mathcal{A} \mid \mathcal{A} \triangleright \mathcal{B};$
$\vdash_{(\mathrm{Id})} \mathcal{A} \triangleright \mathcal{B} \vdash \mathcal{A} \triangleright \mathcal{B} \vdash_{(\mid \triangleright)(\text{X-L})} \mathcal{A} \mid \mathcal{A} \triangleright \mathcal{B} \vdash \mathcal{B};$
$\rceil \vdash_{(\mathrm{Trans})} \mathcal{F} \mid \mathcal{G} \vdash \mathcal{B}$

(Exchange) $\mathcal{F}, \mathcal{A}, \mathcal{B}, \mathcal{G} \vdash_{\mathrm{ILL}} C \vdash_{\mathrm{ILL}} \mathcal{F}, \mathcal{B}, \mathcal{A}, \mathcal{G} \vdash_{\mathrm{ILL}} C$

We show: $((\mathcal{F} \mid \mathcal{A}) \mid \mathcal{B}) \mid \mathcal{G} \vdash C \vdash ((\mathcal{F} \mid \mathcal{B}) \mid \mathcal{A}) \mid \mathcal{G} \vdash C$.

$\lceil \lceil \vdash_{(\mathrm{Id})} \mathcal{F} \vdash \mathcal{F};\ \vdash_{(\mathrm{X} \mid)} \mathcal{B} \mid \mathcal{A} \vdash \mathcal{A} \mid \mathcal{B} \rceil \vdash_{(\mid \vdash)} \mathcal{F} \mid (\mathcal{B} \mid \mathcal{A}) \vdash \mathcal{F} \mid (\mathcal{A} \mid \mathcal{B});$
$((\mathcal{F} \mid \mathcal{A}) \mid \mathcal{B}) \mid \mathcal{G} \vdash C \vdash_{(\mid \triangleright)} (\mathcal{F} \mid \mathcal{A}) \mid \mathcal{B} \vdash \mathcal{G} \triangleright C \vdash_{(\mathrm{A} \mid)(\mathrm{Trans})} \mathcal{F} \mid (\mathcal{A} \mid \mathcal{B}) \vdash \mathcal{G} \triangleright C$
$\rceil \vdash_{(\mathrm{Trans})} \mathcal{F} \mid (\mathcal{B} \mid \mathcal{A}) \vdash \mathcal{G} \triangleright C \vdash_{(\mathrm{A} \mid)(\mathrm{Trans})} (\mathcal{F} \mid \mathcal{B}) \mid \mathcal{A} \vdash \mathcal{G} \triangleright C \vdash_{(\mid \triangleright)} ((\mathcal{F} \mid \mathcal{B}) \mid \mathcal{A}) \mid \mathcal{G} \vdash C$

($\otimes$ R) $\mathcal{F} \vdash_{\mathrm{ILL}} \mathcal{A};\ \mathcal{G} \vdash_{\mathrm{ILL}} \mathcal{B} \vdash_{\mathrm{ILL}} \mathcal{F}, \mathcal{G} \vdash_{\mathrm{ILL}} \mathcal{A} \otimes \mathcal{B}$

We show: $\mathcal{F} \vdash \mathcal{A};\ \mathcal{G} \vdash \mathcal{B} \vdash \mathcal{F} \mid \mathcal{G} \vdash \mathcal{A} \mid \mathcal{B}$. Simply by ($\mid \vdash$).

($\otimes$ L) $\mathcal{F}, \mathcal{A}, \mathcal{B} \vdash_{\mathrm{ILL}} C \vdash_{\mathrm{ILL}} \mathcal{F}, \mathcal{A} \otimes \mathcal{B} \vdash_{\mathrm{ILL}} C$

We show: $(\mathcal{F} \mid \mathcal{A}) \mid \mathcal{B} \vdash C \vdash \mathcal{F} \mid (\mathcal{A} \mid \mathcal{B}) \vdash C$. Simply by (A $\mid$) and (Trans).

(**1** R) $\vdash_{\mathrm{ILL}} \vdash_{\mathrm{ILL}} \mathbf{1}_{\mathrm{ILL}}$

We show: $\vdash \mathbf{0} \vdash \mathbf{0}$. Simply by (Id).

(**1** L) $\mathcal{F} \vdash_{\mathrm{ILL}} \mathcal{A} \vdash_{\mathrm{ILL}} \mathcal{F}, \mathbf{1}_{\mathrm{ILL}} \vdash_{\mathrm{ILL}} \mathcal{A}$

We show: $\mathcal{F} \vdash \mathcal{A} \vdash \mathcal{F} \mid \mathbf{0} \vdash \mathcal{A}$. Simply by ($\mid$ 0) and (Trans).

(& R) $\mathcal{F} \vdash_{\mathrm{ILL}} \mathcal{A};\ \mathcal{F} \vdash_{\mathrm{ILL}} \mathcal{B} \vdash_{\mathrm{ILL}} \mathcal{F} \vdash_{\mathrm{ILL}} \mathcal{A} \,\&\, \mathcal{B}$

We show: $\mathcal{F} \vdash \mathcal{A};\ \mathcal{F} \vdash \mathcal{B} \vdash \mathcal{F} \vdash \mathcal{A} \wedge \mathcal{B}$. Simply by ($\wedge \vdash$) and (C-L).

(& L1) $\mathcal{F}, \mathcal{A} \vdash_{\mathrm{ILL}} C \vdash_{\mathrm{ILL}} \mathcal{F}, \mathcal{A} \,\&\, \mathcal{B} \vdash_{\mathrm{ILL}} C$

We show: $\mathcal{F} \mid \mathcal{A} \vdash C \vdash \mathcal{F} \mid (\mathcal{A} \wedge \mathcal{B}) \vdash C$.

$\lceil \lceil \vdash_{(\mathrm{Id})} \mathcal{F} \vdash \mathcal{F};\ \lceil \vdash_{(\mathrm{Id})} \mathcal{A} \vdash \mathcal{A} \vdash_{(\text{W-L})} \mathcal{A} \wedge \mathcal{B} \vdash \mathcal{A} \rceil \rceil \vdash_{(\mid \vdash)} \mathcal{F} \mid (\mathcal{A} \wedge \mathcal{B}) \vdash \mathcal{F} \mid \mathcal{A};$
$\mathcal{F} \mid \mathcal{A} \vdash C$
$\rceil \vdash_{(\mathrm{Trans})} \mathcal{F} \mid (\mathcal{A} \wedge \mathcal{B}) \vdash C$

(& L2) $\mathcal{F}, \mathcal{B} \vdash_{\mathrm{ILL}} C \ \}_{\mathrm{ILL}}\ \mathcal{F}, \mathcal{A} \,\&\, \mathcal{B} \vdash_{\mathrm{ILL}} C$

We show: $\mathcal{F} \mid \mathcal{B} \vdash C \ \}\ \mathcal{F} \mid (\mathcal{A} \wedge \mathcal{B}) \vdash C$.

$\big(\big(\ \}_{(\mathrm{Id})}\ \mathcal{F} \vdash \mathcal{F};\ \big(\}_{(\mathrm{Id})}\ \mathcal{B} \vdash \mathcal{B}\ \}_{(\mathrm{W\text{-}L})(\mathrm{X\text{-}L})}\ \mathcal{A} \wedge \mathcal{B} \vdash \mathcal{B}\big)\big)\ \}_{(\mid\vdash)}\ \mathcal{F} \mid (\mathcal{A} \wedge \mathcal{B}) \vdash \mathcal{F} \mid \mathcal{B};$
$\mathcal{F} \mid \mathcal{B} \vdash C$
$\big)\ \}_{(\mathrm{Trans})}\ \mathcal{F} \mid (\mathcal{A} \wedge \mathcal{B}) \vdash C$

($\top$ R) $\}_{\mathrm{ILL}}\ \mathcal{F} \vdash_{\mathrm{ILL}} \top_{\mathrm{ILL}}$

We show: $\}\ \mathcal{F} \vdash \mathbf{T}$.

$\}_{(\mathrm{Id})}\ \mathbf{T} \vdash \mathbf{T}\ \}_{(\mathrm{W\text{-}L})}\ \mathbf{T} \wedge \mathcal{F} \vdash \mathbf{T}\ \}_{(\mathrm{X\text{-}L})}\ \mathcal{F} \wedge \mathbf{T} \vdash \mathbf{T}\ \}_{(\mathbf{T})}\ \mathcal{F} \vdash \mathbf{T}$

($\oplus$ R1) $\mathcal{F} \vdash_{\mathrm{ILL}} \mathcal{A}\ \}_{\mathrm{ILL}}\ \mathcal{F} \vdash_{\mathrm{ILL}} \mathcal{A} \oplus \mathcal{B}$

We show: $\mathcal{F} \vdash \mathcal{A}\ \}\ \mathcal{F} \vdash \mathcal{A} \vee \mathcal{B}$. Simply by (W-R) and (X-R).

($\oplus$ R2) $\mathcal{F} \vdash_{\mathrm{ILL}} \mathcal{B}\ \}_{\mathrm{ILL}}\ \mathcal{F} \vdash_{\mathrm{ILL}} \mathcal{A} \oplus \mathcal{B}$

We show: $\mathcal{F} \vdash \mathcal{B}\ \}\ \mathcal{F} \vdash \mathcal{A} \vee \mathcal{B}$. Simply by (W-R).

($\oplus$ L) $\mathcal{F}, \mathcal{A} \vdash_{\mathrm{ILL}} C;\ \mathcal{F}, \mathcal{B} \vdash_{\mathrm{ILL}} C\ \}_{\mathrm{ILL}}\ \mathcal{F}, \mathcal{A} \oplus \mathcal{B} \vdash_{\mathrm{ILL}} C$

We show: $\mathcal{F} \mid \mathcal{A} \vdash C;\ \mathcal{F} \mid \mathcal{B} \vdash C\ \}\ \mathcal{F} \mid (\mathcal{A} \vee \mathcal{B}) \vdash C$.

$\big(\big(\mathcal{F} \mid \mathcal{A} \vdash C;\ \mathcal{F} \mid \mathcal{B} \vdash C\big)\ \}_{(\vee\vdash)}\ (\mathcal{F} \mid \mathcal{A}) \vee (\mathcal{F} \mid \mathcal{B}) \vdash C \vee C;$
$\}_{(\mid\vee)}\ \mathcal{F} \mid (\mathcal{A} \vee \mathcal{B}) \vdash (\mathcal{F} \mid \mathcal{A}) \vee (\mathcal{F} \mid \mathcal{B})$
$\big)\ \}_{(\mathrm{Trans})(\mathrm{C\text{-}R})}\ \mathcal{F} \mid (\mathcal{A} \vee \mathcal{B}) \vdash C$

(**0** L) $\}_{\mathrm{ILL}}\ \mathcal{F}, \mathbf{0}_{\mathrm{ILL}} \vdash_{\mathrm{ILL}} \mathcal{A}$

We show: $\}\ \mathcal{F} \mid \mathbf{F} \vdash \mathcal{A}$.

$\}_{(\mathrm{Id})}\ \mathbf{F} \vdash \mathbf{F}\ \}_{(\mathrm{W\text{-}R})}\ \mathbf{F} \vdash (\mathcal{F} \triangleright \mathcal{A}) \vee \mathbf{F}\ \}_{(\mathrm{X\text{-}R})}\ \mathbf{F} \vdash \mathbf{F} \vee (\mathcal{F} \triangleright \mathcal{A})\ \}_{(\mathbf{F})}\ \mathbf{F} \vdash \mathcal{F} \triangleright \mathcal{A}$
$\}_{(\mid\triangleright)}\ \mathbf{F} \mid \mathcal{F} \vdash \mathcal{A}\ \}_{(\mathrm{X}\mid)(\mathrm{Trans})}\ \mathcal{F} \mid \mathbf{F} \vdash \mathcal{A}$

($\multimap$ R) $\mathcal{F}, \mathcal{A} \vdash_{\mathrm{ILL}} \mathcal{B}\ \}_{\mathrm{ILL}}\ \mathcal{F} \vdash_{\mathrm{ILL}} \mathcal{A} \multimap \mathcal{B}$

We show: $\mathcal{F} \mid \mathcal{A} \vdash \mathcal{B}\ \}\ \mathcal{F} \vdash \mathcal{A} \triangleright \mathcal{B}$. Simply by ($\mid\triangleright$).

($\multimap$ L) $\mathcal{F} \vdash_{\mathrm{ILL}} \mathcal{A};\ \mathcal{G}, \mathcal{B} \vdash_{\mathrm{ILL}} C\ \}_{\mathrm{ILL}}\ \mathcal{F}, \mathcal{G}, \mathcal{A} \multimap \mathcal{B} \vdash_{\mathrm{ILL}} C$

We show: $\mathcal{F} \vdash \mathcal{A};\ \mathcal{G} \mid \mathcal{B} \vdash C\ \}\ \mathcal{F} \mid \mathcal{G} \mid \mathcal{A} \triangleright \mathcal{B} \vdash C$.

$\big(\big(\big(\mathcal{F} \vdash \mathcal{A};\ \}_{(\mathrm{Id})}\ \mathcal{A} \triangleright \mathcal{B} \vdash \mathcal{A} \triangleright \mathcal{B}\big)\ \}_{(\mid\vdash)}\ \mathcal{F} \mid \mathcal{A} \triangleright \mathcal{B} \vdash \mathcal{A} \mid \mathcal{A} \triangleright \mathcal{B};$
$\}_{(\mathrm{Id})}\ \mathcal{A} \triangleright \mathcal{B} \vdash \mathcal{A} \triangleright \mathcal{B}\ \}_{(\mid\triangleright)(\mathrm{X\text{-}L})}\ \mathcal{A} \mid \mathcal{A} \triangleright \mathcal{B} \vdash \mathcal{B};$
$\big)\ \}_{(\mathrm{Trans})}\ \mathcal{F} \mid \mathcal{A} \triangleright \mathcal{B} \vdash \mathcal{B}\ \}_{(\mid\vdash)(\ldots)}\ \mathcal{F} \mid \mathcal{G} \mid \mathcal{A} \triangleright \mathcal{B} \vdash \mathcal{G} \mid \mathcal{B};$
$\mathcal{G} \mid \mathcal{B} \vdash C$
$\big)\ \}_{(\mathrm{Trans})}\ \mathcal{F} \mid \mathcal{G} \mid \mathcal{A} \triangleright \mathcal{B} \vdash C$

(**!** L1) $\}_{\mathrm{ILL}}\ !\mathcal{A} \vdash_{\mathrm{ILL}} \mathbf{1}_{\mathrm{ILL}}$

We show: $\}\ \mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}} \vdash \mathbf{0}$.

$\}_{(\mathrm{Id})}\ \mathbf{0} \vdash \mathbf{0}\ \}_{(\mathrm{W\text{-}L})}\ \ \mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}} \vdash \mathbf{0}$

(**!** L2) $\}_{\mathrm{ILL}}\ !\mathcal{A} \vdash_{\mathrm{ILL}} \mathcal{A}$

We show: $\mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}} \vdash \mathcal{A}$.

$\big(\ \}_{(\mathit{Vld}\,\mathrm{T})}\ (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}} \vdash \mathbf{0} \Rightarrow \mathcal{A}\ \}_{(\mathrm{W\text{-}L})(\mathrm{X\text{-}L})}\ \mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}} \vdash \mathbf{0} \Rightarrow \mathcal{A};$
$\}_{(\mathrm{Id})}\ \mathbf{0} \vdash \mathbf{0}\ \}_{(\mathrm{W\text{-}L})}\ \ \mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}} \vdash \mathbf{0}$

$\rangle \;\downharpoonright_{(\Rightarrow \mathrm{E})}\; \mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}} \vdash \mathcal{A}$

(! L3) $\downharpoonright_{\mathrm{ILL}}\; !\mathcal{A} \vdash_{\mathrm{ILL}} !\mathcal{A} \otimes !\mathcal{A}$

We show: $\downharpoonright\; \mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}} \vdash (\mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}}) \mid (\mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}})$.

First we show that $\mathbf{T} \mid (\mathbf{0} \Rightarrow \mathcal{B}) \vdash \mathbf{0} \Rightarrow \mathcal{B}$.

$\downharpoonright_{(\mid \wedge \mathbf{0})}\; (\mathbf{0} \Rightarrow \mathcal{B}) \mid \mathbf{T} \wedge \mathbf{0} \vdash \mathbf{0} \Rightarrow \mathcal{B}\;\downharpoonright_{(\neg\text{-R})}\; (\mathbf{0} \Rightarrow \mathcal{B}) \mid \mathbf{T} \vdash \neg\mathbf{0} \vee \neg\mathbf{0} \vee \mathcal{B}$

$\downharpoonright_{(\text{A-R})(\text{C-R})(\text{X}\mid)(\text{Trans})}\; \mathbf{T} \mid (\mathbf{0} \Rightarrow \mathcal{B}) \vdash \mathbf{0} \Rightarrow \mathcal{B}$

Now we take $\mathcal{B} = (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}\neg}$:

$\lceil\; \mathbf{T} \mid (\mathbf{0} \Rightarrow (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}\neg}) \vdash \mathbf{0} \Rightarrow (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}\neg}$

$= \mathbf{T} \mid \neg(\mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}}) \vdash \neg(\mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}})$

$\downharpoonright_{(\neg \vdash)(\ldots)}\; \mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}} \vdash \neg(\mathbf{T} \mid \neg(\mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}}))$;

$\downharpoonright_{(\mid \mathbf{T})(\text{X-R})}\; \mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}} \vdash (\mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}}) \mid \mathbf{T}$

$\rangle \;\downharpoonright_{(\wedge \vdash)(\text{C-L})}\; \mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}} \vdash (\mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}}) \mid \mathbf{T} \;\wedge\; \neg(\mathbf{T} \mid \neg(\mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}}))$

$\downharpoonright_{(\mid \neg)}\; (\mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}}) \mid (\mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}})$

(! R) $\mathcal{B} \vdash_{\mathrm{ILL}} \mathbf{1}_{\mathrm{ILL}}$; $\mathcal{B} \vdash_{\mathrm{ILL}} \mathcal{A}$; $\mathcal{B} \vdash_{\mathrm{ILL}} \mathcal{B} \otimes \mathcal{B}\;\downharpoonright_{\mathrm{ILL}}\; \mathcal{B} \vdash_{\mathrm{ILL}} !\mathcal{A}$

We show: $\mathcal{B} \vdash \mathbf{0}$; $\mathcal{B} \vdash \mathcal{A}$; $\mathcal{B} \vdash \mathcal{B} \mid \mathcal{B}\;\downharpoonright\; \mathcal{B} \vdash \mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}}$.

(The assumption $\mathcal{B} \vdash \mathcal{B} \mid \mathcal{B}$ is not used.)

$\lceil\; \lceil\; \lceil\; \lceil\; \mathcal{B} \vdash \mathcal{A}\;\downharpoonright_{(\neg \vdash)}\; \neg\mathcal{A} \vdash \neg\mathcal{B} \rangle\; ;\; \mathcal{B} \vdash \mathbf{0} \rangle\;\downharpoonright_{(\mid \vdash)}\; \neg\mathcal{A} \mid \mathcal{B} \vdash \neg\mathcal{B} \mid \mathbf{0}$

$\downharpoonright_{(\mid \mathbf{0})(\text{Trans})}\; \neg\mathcal{A} \mid \mathcal{B} \vdash \neg\mathcal{B}\;\downharpoonright_{(\neg\text{-L})(\text{X-L})(\ldots)}\; (\mathbf{0} \mid \mathcal{B}) \wedge (\neg\mathcal{A} \mid \mathcal{B}) \vdash \mathbf{F}$;

$\downharpoonright_{(\mid \wedge)}\; (\mathbf{0} \wedge \neg\mathcal{A}) \mid \mathcal{B} \vdash (\mathbf{0} \mid \mathcal{B}) \wedge (\neg\mathcal{A} \mid \mathcal{B})$

$\rangle \;\downharpoonright_{(\text{Trans})}\; (\mathbf{0} \wedge \neg\mathcal{A}) \mid \mathcal{B} \vdash \mathbf{F}\;\downharpoonright_{(\text{X}\mid)}\; \mathcal{B} \mid (\mathbf{0} \wedge \neg\mathcal{A}) \vdash \mathbf{F}$

$\downharpoonright_{(\mid \rhd)}\; \mathcal{B} \vdash (\mathbf{0} \wedge \neg\mathcal{A})^{\mathbf{F}} = (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}}$;

$\mathcal{B} \vdash \mathbf{0}$

$\rangle \;\downharpoonright_{(\wedge \vdash)(\text{C-L})}\; \mathcal{B} \vdash \mathbf{0} \wedge (\mathbf{0} \Rightarrow \mathcal{A})^{\neg\mathbf{F}}$

□

Remark. The rules $\mathcal{T} \vdash_{\mathrm{ILL}} \mathcal{B}\;\downharpoonright_{\mathrm{ILL}}\; \mathcal{T}, \mathcal{A} \vdash_{\mathrm{ILL}} \mathcal{B}$ (weakening) and $\mathcal{T}, \mathcal{A}, \mathcal{A} \vdash_{\mathrm{ILL}} \mathcal{B}\;\downharpoonright_{\mathrm{ILL}}\; \mathcal{T}, \mathcal{A} \vdash_{\mathrm{ILL}} \mathcal{B}$ (contraction) are not valid. Under our interpretation, they mean $\mathcal{T} \vdash \mathcal{B}\;\downharpoonright\; \mathcal{T} \mid \mathcal{A} \vdash \mathcal{B}$ and $\mathcal{T} \mid \mathcal{A} \mid \mathcal{A} \vdash \mathcal{B}\;\downharpoonright\; \mathcal{T} \mid \mathcal{A} \vdash \mathcal{B}$. For weakening, note that $\mathbf{0} \vdash \mathbf{0}$ is valid, but $\mathbf{0} \mid n[\mathbf{T}] \vdash \mathbf{0}$ is not. For contraction, note that $\mathbf{0} \mid n[\mathbf{T}] \mid n[\mathbf{T}] \vdash n[\mathbf{T}] \mid n[\mathbf{T}]$ is valid, but $\mathbf{0} \mid n[\mathbf{T}] \vdash n[\mathbf{T}] \mid n[\mathbf{T}]$ is not. To satisfy weakening, we would need an ambient calculus where at least $n[P] \equiv \mathbf{0}$, but then much would collapse (e.g. $n[P] \equiv m[Q]$). To satisfy contraction, we would need an ambient calculus where at least $n[P] \mid n[Q] \equiv n[P \mid Q]$, so that a process satisfying $n[m[\mathbf{T}]] \mid n[m[\mathbf{T}]]$ could be hierarchically coalesced to satisfy $n[m[\mathbf{T}]]$.

□

8-10 Corollary

Let $\mathcal{A}_1, \ldots, \mathcal{A}_n, \mathcal{B}$ be formulas in ILL.

$\boldsymbol{vld}_{\mathrm{ILL}}(\mathcal{A}_1, \ldots, \mathcal{A}_n \vdash_{\mathrm{ILL}} \mathcal{B}) \;\Rightarrow\; \mathcal{A}_1 \mid \ldots \mid \mathcal{A}_n \vdash \mathcal{B}$.

$\mathcal{A}_1 \mid \ldots \mid \mathcal{A}_n \vdash \mathcal{B} \;\Rightarrow\; \boldsymbol{vld}_{\mathrm{ILL}}(\mathcal{A}_1, \ldots, \mathcal{A}_n \vdash_{\mathrm{ILL}} \mathcal{B})_{\Theta}$.

Proof

$\mathbf{vld}_{\mathrm{ILL}}(\mathcal{A}_1, ..., \mathcal{A}_n \vdash_{\mathrm{ILL}} \mathcal{B}) \Rightarrow \mathcal{A}_1, ..., \mathcal{A}_n \vdash_{\mathrm{ILL}} \mathcal{B}$ by Proposition 8-3; then $\mathcal{A}_1, ..., \mathcal{A}_n \vdash_{\mathrm{ILL}} \mathcal{B} \Rightarrow \mathcal{A}_1 | ... | \mathcal{A}_n \vdash \mathcal{B}$ by Proposition 8-9.

$\mathcal{A}_1 | ... | \mathcal{A}_n \vdash \mathcal{B} \Rightarrow \mathbf{vld}(\mathcal{A}_1 | ... | \mathcal{A}_n \vdash \mathcal{B})$ by the validity of the inference rules of the ambient logic; then $\mathbf{vld}(\mathcal{A}_1 | ... | \mathcal{A}_n \vdash \mathcal{B}) \Rightarrow \mathbf{vld}_{\mathrm{ILL}}(\mathcal{A}_1, ..., \mathcal{A}_n \vdash_{\mathrm{ILL}} \mathcal{B})_\Theta$ by Proposition 8-8.

□

The structure Θ is in fact more than a quantale: it is also a boolean algebra. Hence we have a classical-logic structure as well, given by $\langle \Phi, \subseteq, \bigcup, \bigcap, \Phi\text{-}complement \rangle$.

9 Related Work and Conclusions

We have introduced a logic for describing concurrent processes with restricted names. Most previous logics for concurrency have strived to describe properties that are invariant under some coarse process equivalence, such as bisimulation. In that context, composition and guarantee were anticipated by Dam [19], and the hidden name quantifier was anticipated by Caires [3]; then followed much work on logics for π-calculus (e.g., see [20]).

Because of our original motivation in describing location structures in detail, the properties described by our logic are much finer, and are invariant only up to structural congruence (see also [32] for a characterization). Because of this, our logic is closely related to intuitionistic linear logic and to bunched logics. Our logic is unusual also because it handles variables ranging over a countable universe of names; these variables can be the subject of universal, existential, fresh-name, and hidden-name quantification.

Our logic is built directly out of a process model, so logical soundness is easy to check. Logical completeness is a much more difficult question. In separate work, it is shown that an interesting propositional fragment is complete [7]. We do not have completeness results for the full logic, but we have shown that, in terms of expressiveness, all the theorems of intuitionistic linear logic can be derived. So far, we have mostly tried to discover as many true logical facts as possible, and to minimize the collection of basic rules. We have concentrated in particular on commutation and distribution properties of operators that can be useful in formal proofs.

Fresh-name quantification is modeled after Gabbay and Pitts [23], adapted to our context; it provides logical rules for reasoning abstractly about freshness. Hidden-name quantification is obtained by combining fresh-name quantification with a revelation operator (not a quantifier) for revealing restricted process names. Most novel axioms have to do with revelation; they often reflect and resemble well-known properties of π-calculus restriction.

A model checker is an algorithm that determines the truth of the assertion $P \vDash \mathcal{A}$, given process P and formula $\mathcal{A}$ as input. Model checking the ambient logic is beyond the scope of the present article, but is considered in others. Our conference paper [13] describes a simple algorithm for model checking the fragment of the calculus without replication and restriction against the fragment of the logic without guarantee and revelation. Charatonik and others [15] present a PSPACE algorithm and show the problem to be PSPACE-complete. Charatonik and Talbot [16] show that either including replication in the calculus or guarantee in the logic leads to undecidability. Moreover, they extend the algorithm of

Charatonik and others [15] to include restriction in the calculus and revelation in the logic, while preserving its PSPACE complexity.

Sangiorgi [32] studies the equivalence relation that relates ambient processes if they satisfy the same formulas of the logic. He presents a co-inductive characterization of this relation, and shows it to be nearly identical to structural congruence.

Dal Zilio [18] extends the logic with least and greatest fixpoints as in the propositional μ-calculus. He presents a simple condition for coincidence for least and greatest fixpoints.

Ghelli and one of the authors [9] have developed a spatial fragment of the Ambient Logic into a query language for semistructured data: TQL. Logical equivalences can be used for query transformations. Research continues in using the hidden-name quantifier in this context. Related logics for graphs [8] are also being studied.

Parallel work of one of the authors with Caires [4,5] investigates a logic for π-calculus, along the same general lines as the Ambient Logic. Apart from the differences in the underlying calculi, that work uses a technically different formulation of sequents [5] and handles freshness through a logical quantifier (rather than through an axiom schema as done here). That work handles recursive formulas, and in particular the interaction of recursion with freshness, but does so through a more sophisticated model [4] than the one presented here.

Acknowledgments

Giorgio Ghelli participated in the initial discussions leading to this logic. Barney Hilken, Peter O'Hearn, David Pym, and Glynn Winskel directed us to relevant literature. Gordon Plotkin suggested the new axioms for structural congruence, as studied in [22]. Useful comments were made by Martín Abadi and Todd Knoblock.

10 Appendix

10.1 Logical Corollaries

10-1 Logical Corollaries of Proposition 4-5 (Propositional Logic)

(Ded) $\mathcal{A} \vdash \mathcal{B} \;\Leftrightarrow\; \mathbf{T} \vdash \mathcal{A} \Rightarrow \mathcal{B}$

(Trans) $\mathcal{A} \vdash \mathcal{B} \;\wedge\; \mathcal{B} \vdash C \;\Rightarrow\; \mathcal{A} \vdash C$

(Valid) $\mathcal{A} \vdash \mathcal{B} \;\Rightarrow\; (\mathbf{T} \vdash \mathcal{A} \Rightarrow \mathbf{T} \vdash \mathcal{B})$

($\wedge\vdash$) $\mathcal{A} \vdash \mathcal{B};\; \mathcal{A}' \vdash \mathcal{B}' \;\}\; \mathcal{A} \wedge \mathcal{A}' \vdash \mathcal{B} \wedge \mathcal{B}'$

($\vee\vdash$) $\mathcal{A} \vdash \mathcal{B};\; \mathcal{A}' \vdash \mathcal{B}' \;\}\; \mathcal{A} \vee \mathcal{A}' \vdash \mathcal{B} \vee \mathcal{B}'$

($\neg\vdash$) $\mathcal{B} \vdash \mathcal{A} \;\}\; \neg\mathcal{A} \vdash \neg\mathcal{B}$

($\Rightarrow\vdash$) $\mathcal{B} \vdash \mathcal{A};\; \mathcal{A}' \vdash \mathcal{B}' \;\}\; \mathcal{A} \Rightarrow \mathcal{A}' \vdash \mathcal{B} \Rightarrow \mathcal{B}'$

□

10-2 Logical Corollaries of Proposition 4-6 (Composition Rules)

(| ∧) $\quad \vdash (\mathcal{A} \wedge \mathcal{B}) \mid C \vdash \mathcal{A} \mid C \wedge \mathcal{B} \mid C$

(| ∨) $\quad \vdash \mathcal{A} \mid C \vee \mathcal{B} \mid C \vdash (\mathcal{A} \vee \mathcal{B}) \mid C$

(| **T**) $\quad \vdash \mathcal{A} \vdash \mathbf{T} \mid \mathcal{A}$

(| **F**) $\quad \vdash \mathbf{F} \mid \mathcal{A} \vdash \mathbf{F}$

(**0** ¬ |) $\quad \vdash \mathbf{0} \vdash \neg(\neg\mathbf{0} \mid \neg\mathbf{0})$

(| ∧**0**) $\quad \vdash \mathcal{A} \mid \mathcal{B} \wedge \mathbf{0} \vdash \mathcal{A}$

(| ¬) $\quad \vdash \neg(\mathcal{A} \mid \mathcal{B}) \wedge (\mathcal{A} \mid \mathbf{T}) \vdash (\mathbf{T} \mid \neg\mathcal{B})$

$\quad \vdash \neg(\mathbf{T} \mid \mathcal{B}) \vdash \mathbf{T} \mid \neg\mathcal{B}$

$\quad \vdash \neg(\mathcal{A} \mid \mathcal{B}) \vdash (\neg\mathcal{A} \mid \mathbf{T}) \vee (\mathbf{T} \mid \neg\mathcal{B})$

(|-E) $\quad \mathcal{A} \vdash \mathcal{B}' \mid \mathcal{B}''$; $\mathcal{A}' \wedge (\mathcal{B}' \mid C'') \vdash \mathcal{D}$; $\mathcal{A}'' \wedge (C' \mid \mathcal{B}'') \vdash \mathcal{D}$

$\quad \vdash (\mathcal{A} \wedge (\mathcal{A}' \wedge \mathcal{A}'')) \wedge (C' \parallel C'') \vdash \mathcal{D}$

(▷ ⊢) $\quad \mathcal{A}' \vdash \mathcal{A}$; $\mathcal{B} \vdash \mathcal{B}' \vdash \mathcal{A} \triangleright \mathcal{B} \vdash \mathcal{A}' \triangleright \mathcal{B}'$

(▷**F** ⊢) $\quad \mathcal{B} \vdash \mathcal{A} \vdash \mathcal{A}^{\mathbf{F}} \vdash \mathcal{B}^{\mathbf{F}}$

(▷ |) $\quad \vdash (\mathcal{A} \triangleright \mathcal{B}) \mid \mathcal{A} \vdash \mathcal{B}$

(▷▷) $\quad \vdash (\mathcal{A} \triangleright \mathcal{B}) \mid (\mathcal{B} \triangleright C) \vdash \mathcal{A} \triangleright C$

(▷-L) $\quad \mathcal{D} \vdash \mathcal{A}$; $\mathcal{B} \vdash C \vdash \mathcal{D} \mid (\mathcal{A} \triangleright \mathcal{B}) \vdash C$

□

10-3 Logical Corollaries of Proposition 4-7 (Location Rules)

(n[] ∧) $\quad \vdash x[\mathcal{A} \wedge \mathcal{B}] \vdash x[\mathcal{A}] \wedge x[\mathcal{B}]$

(n[] ∨) $\quad \vdash x[\mathcal{A}] \vee x[\mathcal{B}] \vdash x[\mathcal{A} \vee \mathcal{B}]$

(n[] **F**) $\quad \vdash x[\mathbf{F}] \vdash \mathbf{F}$

(@ ⊢) $\quad \mathcal{A} \vdash \mathcal{B} \vdash \mathcal{A}@x \vdash \mathcal{B}@x$

(n[$\mathcal{A}$@n]) $\quad \vdash x[\mathcal{A}@x] \vdash \mathcal{A}$

(n[$\mathcal{A}$]@n) $\quad \vdash \mathcal{A} \dashv\vdash x[\mathcal{A}]@x$

(n[¬$\mathcal{A}$]) $\quad \vdash x[\mathcal{A}] \vdash \neg x[\neg\mathcal{A}]$; $\quad \vdash x[\neg\mathcal{A}] \vdash \neg x[\mathcal{A}]$

(¬n[$\mathcal{A}$]) $\quad \vdash \neg x[\neg\mathcal{A}] \dashv\vdash x[\mathbf{T}] \Rightarrow x[\mathcal{A}]$; $\quad \vdash \neg x[\mathcal{A}] \dashv\vdash \neg x[\mathbf{T}] \vee x[\neg\mathcal{A}]$

□

The converse of (@ ⊢) is not true. It is equivalent to asking that ($\forall P{:}\Pi.\ n[P] \vDash \mathcal{A} \Rightarrow n[P] \vDash \mathcal{B}) \Rightarrow (\forall P{:}\Pi.\ P \vDash \mathcal{A} \Rightarrow P \vDash \mathcal{B})$. Let $\mathcal{A} = n[\mathbf{T}] \vee \neg n[\mathbf{T}]$ and $\mathcal{B} = n[\mathbf{T}]$. Then ($\forall P{:}\Pi.\ n[P] \vDash (n[\mathbf{T}] \vee \neg n[\mathbf{T}]) \Rightarrow n[P] \vDash n[\mathbf{T}]$), but not ($\forall P{:}\Pi.\ P \vDash (n[\mathbf{T}] \vee \neg n[\mathbf{T}]) \Rightarrow P \vDash n[\mathbf{T}]$) because of, say, $P = \mathbf{0}$.

10-4 Logical Corollaries of Proposition 4-8 (Modalities)

($\Diamond \vdash$) $\mathcal{A} \vdash \mathcal{B} \;\}\; \Diamond\mathcal{A} \vdash \Diamond\mathcal{B}$

($\Box \wedge$) $\}\; \Box(\mathcal{A} \wedge \mathcal{B}) \dashv\vdash \Box\mathcal{A} \wedge \Box\mathcal{B}$

($\Diamond$ T) $\}\; \mathcal{A} \vdash \Diamond\mathcal{A}$

($\Box \Diamond$) $\}\; \Box\mathcal{A} \vdash \Diamond\mathcal{A}$

($\Diamond$ K) $\}\; \Diamond\mathcal{A} \Rightarrow \Diamond\mathcal{B} \vdash \Diamond(\mathcal{A} \Rightarrow \mathcal{B})$

($\Diamond$ 4) $\}\; \Diamond\Diamond\mathcal{A} \vdash \Diamond\mathcal{A}$

($\Diamond \vee$) $\}\; \Diamond(\mathcal{A} \vee \mathcal{B}) \dashv\vdash \Diamond\mathcal{A} \vee \Diamond\mathcal{B}$

($\Diamond$ **F**) $\}\; \Diamond\mathbf{F} \vdash \mathbf{F}$

($\Box \mathbin{\text{¤}}$) $\}\; \Box \mathbin{\text{¤}} \mathcal{A} \vdash \mathbin{\text{¤}} \Box\mathcal{A}$

($\Diamond \vdash$) $\mathcal{A} \vdash \mathcal{B} \;\}\; \Diamond\mathcal{A} \vdash \Diamond\mathcal{B}$

($\mathbin{\text{¤}} \wedge$) $\}\; \mathbin{\text{¤}}(\mathcal{A} \wedge \mathcal{B}) \dashv\vdash \mathbin{\text{¤}}\mathcal{A} \wedge \mathbin{\text{¤}}\mathcal{B}$

($\Diamond$ T) $\}\; \mathcal{A} \vdash \Diamond\mathcal{A}$

($\mathbin{\text{¤}} \Diamond$) $\}\; \mathbin{\text{¤}}\mathcal{A} \vdash \Diamond\mathcal{A}$

($\Diamond$ K) $\}\; \Diamond\mathcal{A} \Rightarrow \Diamond\mathcal{B} \vdash \Diamond(\mathcal{A} \Rightarrow \mathcal{B})$

($\Diamond$ 4) $\}\; \Diamond\Diamond\mathcal{A} \vdash \Diamond\mathcal{A}$

($\Diamond \vee$) $\}\; \Diamond(\mathcal{A} \vee \mathcal{B}) \dashv\vdash \Diamond\mathcal{A} \vee \Diamond\mathcal{B}$

($\Diamond$ **F**) $\}\; \Diamond\mathbf{F} \vdash \mathbf{F}$

□

10-5 Logical Corollaries of Proposition 4-8 (Modalities and Locations 1)

($\Diamond$ @) $\}\; \mathcal{A}@n \vdash (\Diamond\mathcal{A})@n$

($\Box$ @) $\}\; (\Box\mathcal{A})@n \vdash \mathcal{A}@n$

($\Diamond \triangleright$) $\}\; (\Diamond\mathcal{A})\triangleright\mathcal{B} \vdash \mathcal{A}\triangleright\mathcal{B}$; $\}\; (\Diamond\mathcal{A})\triangleright\mathcal{B} \vdash \Diamond(\mathcal{A}\triangleright\mathcal{B})$

($\Box \triangleright$) $\}\; \mathcal{A}\triangleright\mathcal{B} \vdash (\Box\mathcal{A})\triangleright\mathcal{B}$; $\}\; \Box(\mathcal{A}\triangleright\mathcal{B}) \vdash (\Box\mathcal{A})\triangleright\mathcal{B}$

□

The following corollaries depend on the rules ($\Diamond$ *n*[]) and ($\Diamond$ |).

10-6 Logical Corollaries of Proposition 4-9 (Modalities and Locations 2)

($\Diamond$ @-2) $\}\; \Diamond(\mathcal{A}@n) \dashv\vdash (\Diamond\mathcal{A})@n$

($\Diamond \triangleright$-2) $\}\; \Diamond(\mathcal{A}\triangleright\mathcal{B}) \vdash (\Diamond\mathcal{A})\triangleright(\Diamond\mathcal{B})$

($\Box$ *n*[]) $\}\; \Box n[\mathcal{A}] \vdash n[\Box\mathcal{A}]$

□

Remarks.

- The converse of ($\Box$ *n*[]) is not valid: $n[m[\mathit{out}\ n]] \vDash n[\Box\mathbf{T}]$ but $n[m[\mathit{out}\ n]] \nvDash \Box n[\mathbf{T}]$.
- The converse of ($\Diamond$ *n*[]) is not valid: *open m*.*n*[] | *m*[] $\vDash \Diamond n[\mathbf{T}]$ but *open m*.*n*[] | *m*[] $\nvDash n[\Diamond\mathbf{T}]$.
- The rule $\Box(\mathcal{A} \mid \mathcal{B}) \vdash \Box\mathcal{A} \mid \Box\mathcal{B}$ is not valid: $n[m[\mathit{out}\ n]] \vDash \Box(n[\mathbf{T}] \mid \mathbf{T})$ but $n[m[\mathit{out}\ n]] \nvDash \Box n[\mathbf{T}] \mid \Box\mathbf{T}$.

□

10-7 Logical Corollaries of Proposition 4-10 (Satisfiability Rules)

(▷**F** ⊢) $\mathcal{B} \vdash \mathcal{A} \;\}\; \mathcal{A}^{\mathbf{F}} \vdash \mathcal{B}^{\mathbf{F}}$

(1) $\}\; \mathbf{T} \dashv\vdash \mathbf{F}^{\mathbf{F}}$

$\}\; \mathbf{F} \dashv\vdash \mathbf{T}^{\mathbf{F}}$

$\}\; \mathbf{T} \dashv\vdash \mathbf{T}^{\neg\mathbf{F}}$

$\}\; \mathbf{F} \dashv\vdash \mathbf{F}^{\mathbf{F}\neg}$

$\}\; \mathcal{A} \mid \mathcal{A}^{\mathbf{F}} \vdash \mathbf{F}$

$\nvdash \mathcal{A} \vdash \mathcal{A}^{\mathbf{FF}}$
$\nvdash \mathcal{A}^{\neg\mathbf{F}} \vdash \mathcal{A}^{\neg\neg}$; $\nvdash \mathcal{A}^{\mathbf{FF}} \vdash \mathcal{A}^{\mathbf{F}\neg}$; $\nvdash \mathcal{A}^{\neg\mathbf{F}} \vdash \mathcal{A}^{\mathbf{FF}}$; $\nvdash \mathcal{A}^{\neg\neg} \vdash \mathcal{A}^{\mathbf{F}\neg}$; $\nvdash \mathcal{A}^{\neg\mathbf{F}} \vdash \mathcal{A}^{\mathbf{F}\neg}$
$\nvdash \mathcal{A}^{\mathbf{F}\neg} \dashv\vdash \mathcal{A}^{\mathbf{FF}}$

(2) $\mathbf{T} \vdash \mathcal{A}^{\mathbf{F}}$ $\{\!\}$ $\nvdash \mathcal{A} \vdash \mathbf{F}$

(3) $\nvdash \mathcal{B} \triangleright \mathcal{A} \vdash \mathcal{A}^{\mathbf{F}} \triangleright \mathcal{B}^{\mathbf{F}}$

□

10-8 Logical Corollaries of Proposition 4-10 (Modalities and Validity 1)

($\Diamond \triangleright\mathbf{F}$) $\nvdash (\Diamond\mathcal{A})^{\mathbf{F}} \vdash \mathcal{A}^{\mathbf{F}}$; $\nvdash (\Diamond\mathcal{A})^{\mathbf{F}} \vdash \Diamond(\mathcal{A}^{\mathbf{F}})$

($\Box \triangleright\mathbf{F}$) $\nvdash \mathcal{A}^{\mathbf{F}} \vdash (\Box\mathcal{A})^{\mathbf{F}}$; $\nvdash \Box(\mathcal{A}^{\mathbf{F}}) \vdash (\Box\mathcal{A})^{\mathbf{F}}$

□

The following corollaries depend on the rules ($\Diamond\, n[]$) and ($\Diamond$ I).

10-9 Logical Corollaries of Proposition 4-10 (Modalities and Validity 2)

($\Diamond \triangleright\mathbf{F}$-2) $\nvdash \Diamond(\mathcal{A}^{\mathbf{F}}) \dashv\vdash (\Diamond\mathcal{A})^{\mathbf{F}}$; $\nvdash \Diamond(\mathcal{A}^{\mathbf{F}}) \dashv\vdash \mathcal{A}^{\mathbf{F}}$

($\Box \triangleright\mathbf{F}$-2) $\nvdash \Box(\mathcal{A}^{\mathbf{F}}) \vdash (\Box\mathcal{A})^{\mathbf{F}}$; $\nvdash \Box(\mathcal{A}^{\mathbf{F}}) \dashv\vdash \mathcal{A}^{\mathbf{F}}$

□

Remark. The converse of ($\Box \triangleright\mathbf{F}$-2), $(\Box\mathcal{A})^{\mathbf{F}} \vdash \Box(\mathcal{A}^{\mathbf{F}})$ is not valid. It means:

$(\forall P'{:}\Pi.\ \exists P''{:}\Pi.\ P' \rightarrow^{*} P'' \wedge \neg P'' \vDash \mathcal{A}) \Rightarrow (\forall Q{:}\Pi.\ \neg Q \vDash \mathcal{A})$

Its contrapositive is:

$(\exists Q{:}\Pi.\ Q \vDash \mathcal{A}) \Rightarrow (\exists P'{:}\Pi.\ \forall P''{:}\Pi.\ P' \rightarrow^{*} P'' \Rightarrow P'' \vDash \mathcal{A})$

which asserts that if a formula is satisfiable, then there is a process that stably satisfies that formula under all reductions. Consider the formula $n[\mathbf{T}] \wedge \Diamond m[\mathbf{T}]$. This is satisfiable, but once $m[\mathbf{T}]$ holds, $n[\mathbf{T}]$ can no longer hold. □

10-10 Logical Corollaries of Proposition 4-10 (*Vld,Sat* are Modal S5)

(*Sat*) $\nvdash Sat\,\mathcal{A} \dashv\vdash \neg Vld \neg\mathcal{A}$

(*Vld* K) $\nvdash Vld(\mathcal{A} \Rightarrow \mathcal{B}) \vdash (Vld\,\mathcal{A}) \Rightarrow (Vld\,\mathcal{B})$

(*Vld* T) $\nvdash Vld\,\mathcal{A} \vdash \mathcal{A}$

(*Vld* 4) $\nvdash Vld\,\mathcal{A} \vdash Vld\,Vld\,\mathcal{A}$

(*Vld* 5) $\nvdash Sat\,\mathcal{A} \vdash Vld\,Sat\,\mathcal{A}$

(*Vld* $\vdash$) $\mathcal{A} \vdash \mathcal{B}$ $\nvdash Vld\,\mathcal{A} \vdash Vld\,\mathcal{B}$

(*Vld* $\wedge$) $\nvdash Vld(\mathcal{A} \wedge \mathcal{B}) \dashv\vdash Vld\,\mathcal{A} \wedge Vld\,\mathcal{B}$

(*Sat* $\vee$) $\nvdash Sat(\mathcal{A} \vee \mathcal{B}) \dashv\vdash Sat\,\mathcal{A} \vee Sat\,\mathcal{B}$

(*Vld* **T**) $\nvdash \mathbf{T} \vdash Vld\,\mathbf{T}$

(*Sat* **F**) $\nvdash Sat\,\mathbf{F} \vdash \mathbf{F}$

□

From the Logical Corollaries 10-9, we also obtain, for example, $\Diamond\, Vld\,\mathcal{A} \dashv\vdash \Box\, Vld\,\mathcal{A}$.

10-11 Logical Corollaries of Proposition 4-11 (Quantifiers)

(∃-L)	$\mathcal{A} \vdash \mathcal{B} \;\vdash\; \exists x.\mathcal{A} \vdash \mathcal{B}$	where $x \notin fv(\mathcal{B})$
(∃-R)	$\mathcal{A} \vdash \mathcal{B}\{x \leftarrow \eta\} \;\vdash\; \mathcal{A} \vdash \exists x.\mathcal{B}$	where η is a name or a variable
(Subst)	$\mathcal{A} \vdash \mathcal{B} \;\vdash\; \mathcal{A}\{x \leftarrow \eta\} \vdash \mathcal{B}\{x \leftarrow \eta\}$	
(∀-E)	$\mathcal{A} \vdash \forall x.\mathcal{B} \;\vdash\; \mathcal{A} \vdash \mathcal{B}\{x \leftarrow \eta\}$	
(∃-E)	$\mathcal{A} \vdash \exists x.\mathcal{B};\ \mathcal{A} \wedge \mathcal{B} \vdash \mathcal{D} \;\vdash\; \mathcal{A} \vdash \mathcal{D}$	where $x \notin fv(\mathcal{A} \vee \mathcal{D})$
(∀ α)	$\vdash \forall x.\mathcal{A} \dashv\vdash \forall y.\mathcal{A}\{x \leftarrow y\}$	where $y \notin fv(\mathcal{A})$
(∃ α)	$\vdash \exists y.\mathcal{A}\{x \leftarrow y\} \dashv\vdash \exists x.\mathcal{A}$	where $y \notin fv(\mathcal{A})$
(∀ ⊢)	$\mathcal{A} \vdash \mathcal{B} \;\vdash\; \forall x.\mathcal{A} \vdash \forall x.\mathcal{B}$	
(∃ ⊢)	$\mathcal{A} \vdash \mathcal{B} \;\vdash\; \exists x.\mathcal{A} \vdash \exists x.\mathcal{B}$	
(∀ |)	$\vdash (\forall x.\mathcal{A}) \mid (\forall x.\mathcal{B}) \vdash \forall x.(\mathcal{A} \mid \mathcal{B})$	
(∃ |)	$\vdash \exists x.(\mathcal{A} \mid \mathcal{B}) \vdash (\exists x.\mathcal{A}) \mid (\exists x.\mathcal{B})$	
(∀ n[])	$\vdash \eta[\forall x.\mathcal{A}] \vdash \forall x.\eta[\mathcal{A}]$	where $x \neq \eta$
(∃ n[])	$\vdash \exists x.\eta[\mathcal{A}] \dashv\vdash \eta[\exists x.\mathcal{A}]$	where $x \neq \eta$
(∀ ®)	$\vdash \eta \circledR \forall x.\mathcal{A} \vdash \forall x.\eta \circledR \mathcal{A}$	where $x \neq \eta$
(∃ ®)	$\vdash \exists x.\eta \circledR \mathcal{A} \dashv\vdash \eta \circledR \exists x.\mathcal{A}$	where $x \neq \eta$

□

10-12 Logical Corollaries of Proposition 5-1 (Revelation)

(⊘)	$\vdash \mathcal{A} \oslash x \oslash x \dashv\vdash \mathcal{A} \oslash x$	
(⊘ ⊘)	$\vdash \mathcal{A} \oslash y \oslash x \vdash \mathcal{A} \oslash x \oslash y$	
(® ⊘ R)	$\vdash \mathcal{A} \vdash (x \circledR \mathcal{A}) \oslash x$	
	$\vdash x \circledR \mathcal{A} \vdash (x \circledR \mathcal{A}) \oslash x$	
(® ⊘ L)	$\vdash x \circledR (\mathcal{A} \oslash x) \vdash \mathcal{A}$	
	$\vdash x \circledR (\mathcal{A} \oslash x) \vdash \mathcal{A} \oslash x$	
(® ⊘ |)	$\vdash x \circledR ((\mathcal{A} \mid \mathcal{B}) \oslash x) \dashv\vdash x \circledR (\mathcal{A} \oslash x) \mid x \circledR (\mathcal{B} \oslash x)$	
(® | ®)	$\vdash x \circledR (x \circledR \mathcal{A} \mid x \circledR \mathcal{B}) \dashv\vdash x \circledR \mathcal{A} \mid x \circledR \mathcal{B}$	
	$\vdash x \circledR \mathcal{A} \mid x \circledR \mathcal{B} \dashv\vdash (x \circledR \mathcal{A} \mid x \circledR \mathcal{B}) \oslash x$	
(| ® ⊘)	$\vdash x \circledR \mathcal{A} \mid x \circledR (\mathcal{B} \oslash x) \vdash x \circledR (\mathcal{A} \mid \mathcal{B})$	
(⊘ | ®)	$\vdash \mathcal{A} \oslash x \mid x \circledR \mathcal{B} \vdash (\mathcal{A} \mid x \circledR \mathcal{B}) \oslash x$	
(® ∨)	$\vdash x \circledR (\mathcal{A} \vee \mathcal{B}) \dashv\vdash x \circledR \mathcal{A} \vee x \circledR \mathcal{B}$	
(® ∧)	$\vdash x \circledR (\mathcal{A} \wedge \mathcal{B}) \vdash x \circledR \mathcal{A} \wedge x \circledR \mathcal{B}$	hence: $\vdash x \circledR \mathcal{A} \dashv\vdash x \circledR \mathcal{A} \wedge x \circledR \mathbf{T}$
(® **F**)	$\vdash x \circledR \mathbf{F} \vdash \mathbf{F}$	
(⊘ **T**)	$\vdash \mathbf{T} \dashv\vdash \mathbf{T} \oslash x$	
(⊘ **F**)	$\vdash \mathbf{F} \oslash x \dashv\vdash \mathbf{F}$	
(⊘ ∨)	$\vdash (\mathcal{A} \vee \mathcal{B}) \oslash x \dashv\vdash \mathcal{A} \oslash x \vee \mathcal{B} \oslash x$	
(⊘ ∧)	$\vdash (\mathcal{A} \wedge \mathcal{B}) \oslash x \dashv\vdash \mathcal{A} \oslash x \wedge \mathcal{B} \oslash x$	
(⊘ **0**)	$\vdash \mathbf{0} \vdash \mathbf{0} \oslash x$	
(® ¬**0**)	$\vdash x \circledR \neg\mathbf{0} \vdash \neg\mathbf{0}$	
(⊘ |)	$\vdash \mathcal{A} \oslash x \mid x \circledR (\mathcal{B} \oslash x) \vdash (\mathcal{A} \mid \mathcal{B}) \oslash x$	
	$\vdash (\mathcal{A} \mid \mathcal{B}) \oslash x \vdash \mathcal{A} \oslash x \mid \mathcal{B} \oslash x$	

$(\oslash\ \circledR)$ $\quad \vdash \mathcal{A}\oslash x \vdash (x\circledR\mathcal{A})\oslash x$ $\quad$ hence: $\vdash x\circledR(\mathcal{A}\oslash x) \vdash x\circledR\mathcal{A}$

$(\circledR \wedge \oslash)$ $\quad \vdash x\circledR(\mathcal{A} \wedge \mathcal{B}\oslash x) \dashv\vdash x\circledR\mathcal{A} \wedge \mathcal{B}$ $\quad$ hence: $\vdash x\circledR(\mathcal{B}\oslash x) \dashv\vdash x\circledR\mathbf{T} \wedge \mathcal{B}$

$(\circledR \vee \oslash)$ $\quad \vdash x\circledR(\mathcal{A} \vee \mathcal{B}\oslash x) \vdash x\circledR\mathcal{A} \vee \mathcal{B}$

$(\oslash \vdash)$ $\quad \mathcal{A} \vdash \mathcal{B} \ \vdash \mathcal{A}\oslash x \vdash \mathcal{B}\oslash x$

$(\circledR \oslash \mid)$ $\quad \vdash x\circledR((\mathcal{A} \mid \mathcal{B})\oslash x) \dashv\vdash x\circledR(\mathcal{A}\oslash x) \mid x\circledR(\mathcal{B}\oslash x)$

$(\circledR \oslash \mid)$ $\quad \vdash x\circledR((\mathcal{A} \mid \mathcal{B})\oslash x) \vdash (x\circledR\mathcal{A})\oslash x \mid (x\circledR\mathcal{B})\oslash x$

$(\circledR \wedge \mid)$ $\quad \vdash x\circledR\mathbf{T} \wedge (\mathcal{A} \mid \mathcal{B}) \dashv\vdash (x\circledR\mathbf{T} \wedge \mathcal{A}) \mid (x\circledR\mathbf{T} \wedge \mathcal{B})$

$(\circledR \Rightarrow \mid)$ $\quad \vdash (x\circledR\mathbf{T}\Rightarrow\mathcal{A}) \mid (x\circledR\mathbf{T}\Rightarrow\mathcal{B}) \vdash x\circledR\mathbf{T} \Rightarrow (\mathcal{A} \mid \mathcal{B})$

$(\oslash\ n[])$ $\quad \vdash y[\mathcal{A}]\oslash x \dashv\vdash y[\mathcal{A}\oslash x]$ $\quad (x \neq y)$

$(\oslash\ n[])$ $\quad \vdash x[\mathcal{A}]\oslash x \dashv\vdash \mathbf{F}$

$(@\ \circledR)$ $\quad \vdash (x\circledR\mathcal{A})@x \dashv\vdash \mathbf{F}$

$(@\ \circledR \neq)$ $\quad \vdash (x\circledR\mathcal{A})@y \dashv\vdash x\circledR(\mathcal{A}@y)$ $\quad (x \neq y)$

$(\circledR \oslash\ n[])$ $\quad \vdash x\circledR(y[\mathcal{A}]\oslash x) \dashv\vdash y[x\circledR(\mathcal{A}\oslash x)]$ $\quad (x \neq y)$

$(\circledR \wedge n[])$ $\quad \vdash x\circledR\mathbf{T} \wedge y[\mathcal{A}] \dashv\vdash y[x\circledR\mathbf{T} \wedge \mathcal{A}]$ $\quad (x \neq y)$

$(\oslash\ \circledR \neq)$ $\quad \vdash x\circledR(\mathcal{A}\oslash y) \vdash (x\circledR\mathcal{A})\oslash y$

$(\circledR\ \exists)$ $\quad \vdash \exists x.y\circledR\mathcal{A} \dashv\vdash y\circledR\exists x.\mathcal{A}$ $\quad$ where $x \neq y$

$(\circledR\ \forall)$ $\quad \vdash y\circledR\forall x.\mathcal{A} \vdash \forall x.y\circledR\mathcal{A}$ $\quad$ where $x \neq y$

□

Remark. The derived rule $(\circledR\ \neg\mathbf{0})$ says that if we reveal a restricted name and find non-**0**, then the original process is also non-**0**. That is, non-**0**-ness cannot be hidden by restriction. Consider, for example, the process $P = (\nu n)n[]$. Under many standard behavioral equivalences $\approx$ we have $P \approx \mathbf{0}$ [26]. However, we have $P \models n\circledR\neg\mathbf{0}$, and hence by $(\circledR\ \neg\mathbf{0})$, we have that $P \models \neg\mathbf{0}$. This example shows quite clearly that our logic is finer than standard behavioral equivalences, and that it can inspect the structure of restricted processes. □

10-13 Logical Corollaries (Case Analysis)

Let Cl be a classical predicate (typically, Cl is a side condition of the form $x \neq y$).

$(\mathbf{CA} \mid \wedge)$ $\quad \vdash (Cl \wedge \mathcal{A}) \mid (Cl \wedge \mathcal{B}) \dashv\vdash Cl \wedge (\mathcal{A} \mid \mathcal{B})$

$(\mathbf{CA} \mid \Rightarrow)$ $\quad \vdash (Cl \Rightarrow \mathcal{A}) \mid (Cl \Rightarrow \mathcal{B}) \vdash Cl \Rightarrow (\mathcal{A} \mid \mathcal{B})$

$(\mathbf{CA}\ n[] \wedge)$ $\quad \vdash z[Cl \wedge \mathcal{A}] \dashv\vdash Cl \wedge z[\mathcal{A}]$ $\quad$ where z may occur in Cl

$(\mathbf{CA}\ \circledR \wedge)$ $\quad \vdash z\circledR(Cl \wedge \mathcal{A}) \dashv\vdash Cl \wedge z\circledR\mathcal{A}$ $\quad$ where z may occur in Cl

□

The following rules for $\mathsf{H}x.\mathcal{A}$ can be derived by combining the rules for revelation and for the fresh-name quantifier.

10-14 Logical Corollaries for Definition 7-1 (Hidden-Name Quantifier)

$(\mathsf{H}\ \forall)$ $\quad \vdash \forall x.\ (x\#N \wedge x\circledR\mathbf{T}) \Rightarrow x\circledR\mathcal{A} \dashv\vdash \mathsf{H}x.\mathcal{A}$ $\quad$ where $N \supseteq fnv(\mathcal{A})\text{-}\{x\}$ and $x \notin N$

$(\mathsf{H}\ \exists)$ $\quad \vdash \mathsf{H}x.\mathcal{A} \dashv\vdash \exists x.\ x\#N \wedge x\circledR\mathbf{T} \wedge x\circledR\mathcal{A}$ $\quad$ where $N \supseteq fnv(\mathcal{A})\text{-}\{x\}$ and $x \notin N$

$\dashv\vdash \exists x.\ x\#N \wedge x\circledR\mathcal{A}$

$(\nu\ \mathrm{R})$ $\quad \mathcal{A} \wedge x\#N \wedge x\circledR\mathbf{T} \vdash x\circledR\mathcal{B} \ \vdash \mathcal{A} \vdash \mathsf{H}x.\mathcal{B}$ $\quad$ where $N \supseteq fnv(\mathcal{B})\text{-}\{x\}$ and $x \notin N \cup fv(\mathcal{A})$

(ν L)	$x \circledR \mathcal{A} \wedge x\#N \vdash \mathcal{B} \;\}\; \mathsf{H}x.\mathcal{A} \vdash \mathcal{B}$	where $N \supseteq fnv(\mathcal{A})\text{-}\{x\}$ and $x \notin N \cup fv(\mathcal{B})$
(ν E)	$\mathcal{A} \vdash \mathsf{H}x.\mathcal{B};\; x \circledR \mathcal{B} \wedge x\#N \vdash C \;\}\; \mathcal{A} \vdash C$	where $N \supseteq fnv(\mathcal{B})\text{-}\{x\}$ and $x \notin N \cup fv(C)$
(H ⊢)	$\mathcal{A} \vdash \mathcal{B} \;\}\; \mathsf{H}x.\mathcal{A} \vdash \mathsf{H}x.\mathcal{B}$	
(H *fv*)	$\}\; \mathsf{H}x.(\mathcal{A} \oslash x) \dashv\vdash \mathcal{A}$	where $x \notin fv(\mathcal{A})$
(H *fv*)	$\}\; \mathcal{A} \vdash \mathsf{H}x.\mathcal{A}$	where $x \notin fv(\mathcal{A})$
(H ®)	$\}\; \mathsf{H}x.(x \circledR \mathcal{A}) \dashv\vdash \mathsf{H}x.\mathcal{A}$	
(H ⊘)	$\}\; \mathsf{H}x.(\mathcal{A} \oslash x) \vdash \mathsf{H}x.\mathcal{A}$	
(H **0**)	$\}\; \mathsf{H}x.\mathbf{0} \dashv\vdash \mathbf{0}$	
(H *n*[])	$\}\; \mathsf{H}x.y[\mathcal{A}] \dashv\vdash y[\mathsf{H}x.\mathcal{A}]$	where $x \neq y$
(H ∣)	$\}\; \mathsf{H}x.(\mathcal{A} \mid x \circledR \mathcal{B}) \dashv\vdash (\mathsf{H}x.\mathcal{A}) \mid (\mathsf{H}x.\mathcal{B})$	
(H ⊘ ∣)	$\}\; \mathsf{H}x.(\mathcal{A} \oslash x) \mid \mathsf{H}x.(\mathcal{B} \oslash x) \dashv\vdash \mathsf{H}x.((\mathcal{A} \mid \mathcal{B}) \oslash x)$	

□

10.2 Proofs

Proof of Lemma 3-3

The proof is an extension of the proof of the analogous property for our earlier modal logic [13]. If $m{=}m'$ the lemma holds trivially, so we may assume that $m{\neq}m'$. The proof is by induction on the number of symbols in the closed formula $\mathcal{A}$. The number of symbols in a formula is unchanged by substituting a name for a variable or another name. Consider an arbitrary process P, and any names m and m'. We show only the cases for revelation and hiding. The cases for the other constructs are the same as in our earlier proof.

In the case for revelation, $\mathcal{A} = n \circledR \mathcal{B}$, we prove each direction of the following separately, assuming that $m' \notin fn(P) \cup fn(n \circledR \mathcal{B})$ (and hence $m' \neq n$).

$$P \models n \circledR \mathcal{B} \Leftrightarrow P\{m \leftarrow m'\} \models (n \circledR \mathcal{B})\{m \leftarrow m'\}.$$

(⇒) Assume $P \models n \circledR \mathcal{B}$, that is, there is P' such that $P \equiv (\nu n)P'$ and $P' \models \mathcal{B}$. By Lemma 2-1(1), $P \equiv (\nu n)P'$ implies $n \notin fn(P)$ and $m' \notin fn(P')$. By Lemma 3-2, $m' \notin fn(P)$ and $P \equiv (\nu n)P'$ implies $P\{m \leftarrow m'\} \equiv ((\nu n)P')\{m \leftarrow m'\}$. Since $m' \notin fn(P') \cup fn(\mathcal{B})$, the induction hypothesis implies that $P'\{m \leftarrow m'\} \models \mathcal{B}\{m \leftarrow m'\}$.

Next, suppose that $m{=}n$. We get that $(\nu m')(P'\{m \leftarrow m'\}) \models m' \circledR (\mathcal{B}\{m \leftarrow m'\})$. Since $m' \notin fn(P')$ and $m{=}n$ we have $(\nu m')(P'\{m \leftarrow m'\}) = (\nu m)P' = ((\nu n)P')\{m \leftarrow m'\}$. Moreover, $m' \circledR (\mathcal{B}\{m \leftarrow m'\}) = (n \circledR \mathcal{B})\{m \leftarrow m'\}$. Hence, $((\nu n)P')\{m \leftarrow m'\} \models (n \circledR \mathcal{B})\{m \leftarrow m'\}$. By Lemma 3-1, $P\{m \leftarrow m'\} \equiv ((\nu n)P')\{m \leftarrow m'\}$ implies $P\{m \leftarrow m'\} \models (n \circledR \mathcal{B})\{m \leftarrow m'\}$.

Otherwise, suppose that $m{\neq}n$. We get that $(\nu n)(P'\{m \leftarrow m'\}) \models n \circledR (\mathcal{B}\{m \leftarrow m'\})$. Since $m{\neq}n$ and $m'{\neq}n$ we have $(\nu n)(P'\{m \leftarrow m'\}) = ((\nu n)P')\{m \leftarrow m'\}$. Since $m{\neq}n$ we have $n \circledR (\mathcal{B}\{m \leftarrow m'\}) = (n \circledR \mathcal{B})\{m \leftarrow m'\}$. Hence, $((\nu n)P')\{m \leftarrow m'\} \models (n \circledR \mathcal{B})\{m \leftarrow m'\}$. By Lemma 3-1, $P\{m \leftarrow m'\} \equiv ((\nu n)P')\{m \leftarrow m'\}$ implies $P\{m \leftarrow m'\} \models (n \circledR \mathcal{B})\{m \leftarrow m'\}$.

($\Leftarrow$) Assume $P\{m\leftarrow m'\} \vDash (n)\mathcal{B})\{m\leftarrow m'\}$.

First, suppose that $m=n$. By assumption, there is P' such that $P\{m\leftarrow m'\} \equiv (\nu m')P'$ and $P' \vDash \mathcal{B}\{m\leftarrow m'\}$. By Lemma 2-1(1), $P\{m\leftarrow m'\} \equiv (\nu m')P'$ implies that $m' \notin fn(P\{m\leftarrow m'\})$ and $m \notin fn((\nu m')P')$. Since $m\neq m'$ we get that $m \notin fn(P)\cup fn(P')$ and $P \equiv (\nu m')P'$. By induction hypothesis, $m \notin fn(P')\cup fn(\mathcal{B}\{m\leftarrow m'\})$ implies that $P'\{m'\leftarrow m\} \vDash \mathcal{B}\{m\leftarrow m'\}\{m'\leftarrow m\}$, that is, $P'\{m'\leftarrow m\} \vDash \mathcal{B}$. By definition of satisfaction, $(\nu m)(P'\{m'\leftarrow m\}) \vDash m)\mathcal{B}$, that is, $(\nu m')P' \vDash m)\mathcal{B}$. By Lemma 3-1, $P \equiv (\nu m')P'$ implies $P \vDash m)\mathcal{B}$, that is, $P \vDash n)\mathcal{B}$.

Second, suppose that $m\neq n$. By assumption, there is P' such that $P\{m\leftarrow m'\} \equiv (\nu n)P'$ and $P' \vDash \mathcal{B}\{m\leftarrow m'\}$. By Lemma 2-1(1), $P\{m\leftarrow m'\} \equiv (\nu n)P'$ implies that $m \notin fn((\nu n)P')$. Since $m\neq n$, $m \notin fn(P')$. By induction hypothesis, $m \notin fn(P')\cup fn(\mathcal{B}\{m\leftarrow m'\})$ implies that $P'\{m'\leftarrow m\} \vDash \mathcal{B}\{m\leftarrow m'\}\{m'\leftarrow m\}$, that is, $P'\{m'\leftarrow m\} \vDash \mathcal{B}$. By definition of satisfaction, $(\nu n)(P'\{m'\leftarrow m\}) \vDash n)\mathcal{B}$. By Lemma 3-2, $m' \notin fn(P\{m\leftarrow m'\})$ and $P\{m\leftarrow m'\}\{m'\leftarrow m\} \equiv ((\nu n)P')\{m'\leftarrow m\}$ implies $P\{m\leftarrow m'\}\{m'\leftarrow m\} \equiv ((\nu n)P')\{m'\leftarrow m\}$. Hence from $m' \notin fn(P)$, $m\neq n$, and $n\neq m'$ we can calculate $P = P\{m\leftarrow m'\}\{m'\leftarrow m\} \equiv ((\nu n)P')\{m'\leftarrow m\} = (\nu n)(P'\{m'\leftarrow m\})$. By Lemma 3-1, this and $(\nu n)(P'\{m'\leftarrow m\}) \vDash n)\mathcal{B}$ imply $P \vDash n)\mathcal{B}$.

In the case for hiding, $\mathcal{A} = \mathcal{B}\oslash n$, we prove the following directly, where $m' \notin fn(P)\cup fn(\mathcal{B}\oslash n)$ (and hence $m'\neq n$).

$$P \vDash \mathcal{B}\oslash n \Leftrightarrow P\{m\leftarrow m'\} \vDash (\mathcal{B}\oslash n)\{m\leftarrow m'\}.$$

First, suppose that $m=n$. By definition, $P \vDash \mathcal{B}\oslash n \Leftrightarrow (\nu n)P \vDash \mathcal{B}$. By induction hypothesis, $m' \notin fn((\nu n)P)\cup fn(\mathcal{B})$ implies $(\nu n)P \vDash \mathcal{B} \Leftrightarrow ((\nu n)P)\{n\leftarrow m'\} \vDash \mathcal{B}\{n\leftarrow m'\}$. We have $((\nu n)P)\{n\leftarrow m'\} = (\nu n)P = (\nu m')(P\{m\leftarrow m'\})$, since $m' \notin fn(P)$ and $m=n$. By definition, $(\nu m')(P\{m\leftarrow m'\}) \vDash \mathcal{B}\{n\leftarrow m'\} \Leftrightarrow P\{m\leftarrow m'\} \vDash \mathcal{B}\{n\leftarrow m'\}\oslash m'$. From $m=n$ we have $\mathcal{B}\{n\leftarrow m'\}\oslash m' = (\mathcal{B}\oslash n)\{m\leftarrow m'\}$.

Second, suppose that $m\neq n$. By definition, $P \vDash \mathcal{B}\oslash n \Leftrightarrow (\nu n)P \vDash \mathcal{B}$. By induction hypothesis, $m' \notin fn((\nu n)P)\cup fn(\mathcal{B})$ implies $(\nu n)P \vDash \mathcal{B} \Leftrightarrow ((\nu n)P)\{m\leftarrow m'\} \vDash \mathcal{B}\{m\leftarrow m'\}$. We have $((\nu n)P)\{m\leftarrow m'\} = (\nu n)(P\{m\leftarrow m'\})$, since $m\neq n$ and $m'\neq n$. By definition, $(\nu n)(P\{m\leftarrow m'\}) \vDash \mathcal{B}\{m\leftarrow m'\} \Leftrightarrow P\{m\leftarrow m'\} \vDash \mathcal{B}\{m\leftarrow m'\}\oslash n$. From $m\neq n$ we have $\mathcal{B}\{m\leftarrow m'\}\oslash n = (\mathcal{B}\oslash n)\{m\leftarrow m'\}$.

□

Proof of Proposition 4-1

(1) Assume ***vld***$(\mathcal{A})$; that is, assume $\forall\varphi\in fv(\mathcal{A})\rightarrow\Lambda.\ \forall P\in\Pi.\ P \vDash \mathcal{A}_\varphi$. Take any $\psi\in fv((\mathcal{A})\{x\leftarrow n\})\rightarrow\Lambda$ and any $P\in\Pi$. If $x\in fv(\mathcal{A})$, by instantiating the assumption with $\varphi = \psi\{x\leftarrow n\}$ we have $P \vDash \mathcal{A}_{\psi\{x\leftarrow n\}}$, which is the same as $\mathcal{A}\{x\leftarrow n\}_\psi$, since $x\notin dom(\psi)$. If $x\notin fv(\mathcal{A})$, by instantiating the first assumption with $\varphi = \psi$ we have $P \vDash \mathcal{A}_\psi$, which is the same as $\mathcal{A}\{x\leftarrow n\}_\psi$. In both cases, we have shown that $\forall\psi\in fv((\mathcal{A})\{x\leftarrow n\})\rightarrow\Lambda.\ \forall P\in\Pi.\ P \vDash \mathcal{A}\{x\leftarrow n\}_\psi$, that is, ***vld***$(\mathcal{A}\{x\leftarrow n\})$.

(2) Let $\mathcal{S} = \mathcal{A} \vdash \mathcal{B}$. Assume $\mathcal{S}$ is valid, that is ***vld***$(\mathcal{A} \Rightarrow \mathcal{B})$. By (1), we have ***vld***$((\mathcal{A} \Rightarrow \mathcal{B})\{x \leftarrow n\})$, that is $\mathcal{S}\{x \leftarrow n\}$.

□

Proof of Proposition 4-2

(1) (Proof outline) By induction on the structure of $\mathcal{B}\{-\}$, with induction hypothesis:
$\forall \mathcal{A}', \mathcal{A}'' \in \Phi.$ ***vld***$(\mathcal{A}' \Leftrightarrow \mathcal{A}'') \Rightarrow$ ***vld***$(\mathcal{B}\{\mathcal{A}'\} \Leftrightarrow \mathcal{B}\{\mathcal{A}''\})$, that is:
$\forall \mathcal{A}', \mathcal{A}'' \in \Phi.\ (\forall \varphi \in fv(\mathcal{A}', \mathcal{A}'') \rightarrow \Lambda.\ \forall P \in \Pi.\ P \vDash \mathcal{A}'_\varphi \Leftrightarrow P \vDash \mathcal{A}''_\varphi)$
$\Rightarrow (\forall \varphi \in fv(\mathcal{B}\{\mathcal{A}'\}, \mathcal{B}\{\mathcal{A}''\}) \rightarrow \Lambda.\ \forall P \in \Pi.\ P \vDash \mathcal{B}\{\mathcal{A}'\}_\varphi \Leftrightarrow P \vDash \mathcal{B}\{\mathcal{A}''\}_\varphi)$.

(2) Assume $\mathcal{A}' \dashv\vdash \mathcal{A}''$, that is ***vld***$(\mathcal{A}' \Leftrightarrow \mathcal{A}'')$. By (1), ***vld***$(\mathcal{B}\{\mathcal{A}'\} \Leftrightarrow \mathcal{B}\{\mathcal{A}''\})$, that is $\mathcal{B}\{\mathcal{A}'\} \dashv\vdash \mathcal{B}\{\mathcal{A}''\}$.

□

Proof of Proposition 4-4

(1) Assume ***vld***$(\mathcal{B}\{\mathbf{T}\}) \wedge$ ***vld***$(\mathcal{B}\{\mathbf{F}\})$. Take any $\varphi \in fv(\mathcal{B}\{\mathcal{A}\}) \rightarrow \Lambda$ and $P \in \Pi$. By assumption we have $P \vDash \mathcal{B}_\varphi\{\mathbf{T}\}$ and $P \vDash \mathcal{B}_\varphi\{\mathbf{F}\}$. Since $\mathcal{A}$ is classical, we have also that $\{Q \parallel Q \vDash \mathcal{A}_\varphi\} \in \{\Pi, \emptyset\}$. Consider the case where $\{Q \parallel Q \vDash \mathcal{A}_\varphi\} = \Pi$. For the closed formula $\mathcal{A}_\varphi$ we have ***vld***$(\mathcal{A}_\varphi \Leftrightarrow \mathbf{T})$. By Proposition 4-2, ***vld***$(\mathcal{B}\{\mathcal{A}_\varphi\} \Leftrightarrow \mathcal{B}\{\mathbf{T}\})$, and in particular $P \vDash \mathcal{B}_\varphi\{\mathcal{A}_\varphi\}$ iff $P \vDash \mathcal{B}_\varphi\{\mathbf{T}\}$, hence we obtain $P \vDash \mathcal{B}_\varphi\{\mathcal{A}_\varphi\}$, that is $P \vDash \mathcal{B}\{\mathcal{A}\}_\varphi$. Consider now the case where $\{Q \parallel Q \vDash \mathcal{A}_\varphi\} = \emptyset$. For the closed formula $\mathcal{A}_\varphi$ we have ***vld***$(\mathcal{A}_\varphi \Leftrightarrow \mathbf{F})$. By Proposition 4-2, ***vld***$(\mathcal{B}\{\mathcal{A}_\varphi\} \Leftrightarrow \mathcal{B}\{\mathbf{F}\})$, and in particular $P \vDash \mathcal{B}_\varphi\{\mathcal{A}_\varphi\}$ iff $P \vDash \mathcal{B}_\varphi\{\mathbf{F}\}$, hence we obtain $P \vDash \mathcal{B}_\varphi\{\mathcal{A}_\varphi\}$, that is $P \vDash \mathcal{B}\{\mathcal{A}\}_\varphi$. In both cases, we have shown that $\forall \varphi \in fv(\mathcal{B}\{\mathcal{A}\}) \rightarrow \Lambda.\ \forall P \in \Pi.\ P \vDash \mathcal{B}\{\mathcal{A}\}_\varphi$, that is ***vld***$(\mathcal{B}\{\mathcal{A}\})$.

(2) Let $\mathcal{S}\{-\}$ be $\mathcal{B}'\{-\} \vdash \mathcal{B}''\{-\}$. Assume $\mathcal{S}\{\mathbf{T}\}$ and $\mathcal{S}\{\mathbf{F}\}$, that is ***vld***$((\mathcal{B}' \Rightarrow \mathcal{B}'')\{\mathbf{T}\})$ and ***vld***$((\mathcal{B}' \Rightarrow \mathcal{B}'')\{\mathbf{F}\})$. By (1), we have ***vld***$((\mathcal{B}' \Rightarrow \mathcal{B}'')\{\mathcal{A}\})$, that is $\mathcal{S}\{\mathcal{A}\}$.

□

Proof of Proposition 4-5

(Cut) Assume $\forall P{:}\Pi.\ P \vDash \mathcal{A} \Rightarrow P \vDash C \vee P \vDash \mathcal{B}$, and $\forall P{:}\Pi.\ P \vDash \mathcal{A}' \wedge P \vDash C \Rightarrow P \vDash \mathcal{B}'$. Take any $P{:}\Pi$ and assume $P \vDash \mathcal{A} \wedge P \vDash \mathcal{A}'$. From the first assumption we obtain $P \vDash C \vee P \vDash \mathcal{B}$. That is, either $P \vDash \mathcal{B}$ holds, and then also $P \vDash \mathcal{B} \vee P \vDash \mathcal{B}'$ holds, or $P \vDash C$ holds, in which case from the second assumption we obtain that $P \vDash \mathcal{B}'$ holds, and then also $P \vDash \mathcal{B} \vee P \vDash \mathcal{B}'$ holds. We have shown that $\forall P{:}\Pi.\ P \vDash \mathcal{A} \wedge P \vDash \mathcal{A}' \Rightarrow P \vDash \mathcal{B} \vee P \vDash \mathcal{B}'$.

(All others) Follow by trivial arguments from the definition of sequent validity.

□

Proof of Corollary 6-2

(1)

Case $\Rightarrow$) Assume $\exists m{\in}\Lambda.\ m{\notin}\ fn(P,\mathcal{A}) \wedge P \vDash \mathcal{A}\{x{\leftarrow}m\}$. Take $N = fn(P,\mathcal{A})$ to obtain $\exists N{\in} Fin(\Lambda).\ N \supseteq fn(P,\mathcal{A}) \wedge \exists m{\in}\Lambda.\ m{\notin}N \wedge P \vDash \mathcal{A}\{x{\leftarrow}m\}$.

Case $\Leftarrow$) Assume $\exists N{\in}Fin(\Lambda).\ N \supseteq fn(P,\mathcal{A}) \wedge \exists m{\in}\Lambda.\ m{\notin}N \wedge P \vDash \mathcal{A}\{x{\leftarrow}m\}$. In particular, $\exists m{\in}\Lambda.\ m{\notin}\ fn(P,\mathcal{A}) \wedge P \vDash \mathcal{A}\{x{\leftarrow}m\}$.

(2) Assume $\exists m{\in}\Lambda.\ m{\notin}\ fn(P,\mathcal{A}) \wedge P \vDash \mathcal{A}\{x{\leftarrow}m\}$. By (1), this is equivalent to $\exists N{\in}Fin(\Lambda).\ N \supseteq fn(P,\mathcal{A}) \wedge \exists m{\in}\Lambda.\ m{\notin}N \wedge P \vDash \mathcal{A}\{x{\leftarrow}m\}$. By Proposition 6-1, this is in turn equivalent to $\exists N{\in}Fin(\Lambda).\ N \supseteq fn(P,\mathcal{A}) \wedge \forall m{\in}\Lambda.\ m{\notin}N \Rightarrow P \vDash \mathcal{A}\{x{\leftarrow}m\}$.

(3)

Case $\Rightarrow$) Assume $\exists m{\in}\Lambda.\ m{\notin}fn(P,\mathcal{A}) \wedge P \vDash \mathcal{A}\{x{\leftarrow}m\}$. Take any $N \supseteq fn(P,\mathcal{A})$. From the assumption, by Proposition 6-1, we have $\forall m{\in}\Lambda.\ m{\notin}fn(P,\mathcal{A}) \Rightarrow P \vDash \mathcal{A}\{x{\leftarrow}m\}$. In particular, $\forall m{\in}\Lambda.\ m{\notin}N \Rightarrow P \vDash \mathcal{A}\{x{\leftarrow}m\}$. Then, by Proposition 6-1, $\exists m{\in}\Lambda.\ m{\notin}N \wedge P \vDash \mathcal{A}\{x{\leftarrow}m\}$. We have shown $\forall N{\in}Fin(\Lambda).\ N \supseteq fn(P,\mathcal{A}) \Rightarrow \exists m{\in}\Lambda.\ m{\notin}N \wedge P \vDash \mathcal{A}\{x{\leftarrow}m\}$.

Case $\Leftarrow$) Assume $\forall N{\in}Fin(\Lambda).\ N \supseteq fn(P,\mathcal{A}) \Rightarrow \exists m{\in}\Lambda.\ m{\notin}N \wedge P \vDash \mathcal{A}\{x{\leftarrow}m\}$. Take $N = fn(P,\mathcal{A})$ to obtain $\exists N{\in}Fin(\Lambda).\ N \supseteq fn(P,\mathcal{A}) \wedge \exists m{\in}\Lambda.\ m{\notin}N \wedge P \vDash \mathcal{A}\{x{\leftarrow}m\}$. By (1), $\exists m{\in}\Lambda.\ m{\notin}fn(P,\mathcal{A}) \wedge P \vDash \mathcal{A}\{x{\leftarrow}m\}$.

(4)

Case $\Rightarrow$) Assume $\exists m{\in}\Lambda.\ m{\notin}fn(P,\mathcal{A}) \wedge P \vDash \mathcal{A}\{x{\leftarrow}m\}$. By (3), $\forall N{\in}Fin(\Lambda).\ N \supseteq fn(P,\mathcal{A}) \Rightarrow \exists m{\in}\Lambda.\ m{\notin}N \wedge P \vDash \mathcal{A}\{x{\leftarrow}m\}$. Take any $N{\in}Fin(\Lambda)$ such that $N \supseteq fn(P,\mathcal{A})$; by assumption $\exists m{\in}\Lambda.\ m{\notin}N \wedge P \vDash \mathcal{A}\{x{\leftarrow}m\}$, and by Proposition 6-1, $\forall m{\in}\Lambda.\ m{\notin}N \Rightarrow P \vDash \mathcal{A}\{x{\leftarrow}m\}$. We have shown $\forall N{\in}Fin(\Lambda).\ N \supseteq fn(P,\mathcal{A}) \Rightarrow \forall m{\in}\Lambda.\ m{\notin}N \Rightarrow P \vDash \mathcal{A}\{x{\leftarrow}m\}$.

Case $\Leftarrow$) Assume $\forall N{\in}Fin(\Lambda).\ N \supseteq fn(P,\mathcal{A}) \Rightarrow \forall m{\in}\Lambda.\ m{\notin}N \Rightarrow P \vDash \mathcal{A}\{x{\leftarrow}m\}$. Take $N = fn(P,\mathcal{A})$ to obtain $\exists N{\in}Fin(\Lambda).\ N \supseteq fn(P,\mathcal{A}) \wedge \forall m{\in}\Lambda.\ m{\notin}N \Rightarrow P \vDash \mathcal{A}\{x{\leftarrow}m\}$. By (2), $\exists m{\in}\Lambda.\ m{\notin}fn(P,\mathcal{A}) \wedge P \vDash \mathcal{A}\{x{\leftarrow}m\}$.

□

References

[1] M. Abadi: **Secrecy by Typing in Security Protocols**. Journal of the ACM 46, 5 (September 1999), 749-786.

[2] M. Abadi and A.D. Gordon: **A Calculus for Cryptographic Protocols: the Spi Calculus**. Information and Computation 148(1999):1-70.

[3] L. Caires: **A Model for Declarative Programming and Specification with Concurrency and Mobility**. Ph.D. Thesis, Dept. de Informática, FTC, Universidade Nova de Lisboa, 1999.

[4] L. Caires, L Cardelli: **A Spatial Logic for Concurrency: Part I**. Proceedings TACS 2001, Naoki Kobayashi and Benjamin C. Pierce (Eds.). Lecture Notes in Computer Science vol. 2215. Springer, 2001, pp 1-37.

[5] L. Caires, L Cardelli: **A Spatial Logic for Concurrency: Part II**. L Brim et al. (Eds.) Proceedings CONCUR'02. LNCS 2421, Springer 2002. pp 209-225.

[6] L. Caires, L. Monteiro: **Verifiable and Executable Logic Specifications of Concurrent Ob-**

jects in $\mathcal{L}_\pi$. In Programming Languages and Systems, Proceedings of ESOP'98, Chris Hankin (Ed.), Lecture Notes in Computer Science vol. 1877, Springer, 1998. pp 42-56.

[7] C. Calcagno, L. Cardelli, A.D. Gordon: **Deciding Validity in a Spatial Logic for Trees**. Proceedings of the ACM SIGPLAN Workshop on Types in Language Design and Implementation (TLDI), 2003. pp 62-73.

[8] L. Cardelli, P. Gardner, G. Ghelli: **A Spatial Logic for Querying Graphs**. Proceedings ICALP'02, Peter Widmayer et al. (Eds.). Lecture Notes in Computer Science vol 2380, Springer, 2002. pp 597-610.

[9] L. Cardelli, G. Ghelli: **A Query Language Based on the Ambient Logic**. Proceedings ESOP'01, David Sands (Ed.). Lecture Notes in Computer Science vol. 2028, Springer, 2001, pp 1-22.

[10] L. Cardelli, G. Ghelli, A.D. Gordon: **Types for the Ambient Calculus.** Information and Computation 177:160-194 (2002).

[11] L. Cardelli, G. Ghelli, A.D. Gordon: **Secrecy and Group Creation**. Catuscia Palamidessi, editor. Proceedings of CONCUR 2000. Lecture Notes in Computer Science, Vol. 1877, Springer 2000, pp 365-379.

[12] L. Cardelli, A.D. Gordon: **Mobile Ambients**. Theoretical Computer Science 240:177-213 (2000).

[13] L. Cardelli, A.D. Gordon: **Anytime, Anywhere. Modal Logics for Mobile Ambients.** Proceedings of the 27th ACM Symposium on Principles of Programming Languages, 2000, pp 365-377.

[14] L. Cardelli, A.D. Gordon: **Logical Properties of Name Restriction.** Samson Abramsky (Ed.). Typed Lambda Calculi and Applications. 5th International Conference, TLCA 2001. LNCS 2044, 46-60, Springer, 2001.

[15] W. Charatonik, S. Dal Zilio, A.D. Gordon, S. Mukhopadhyay, and J.-M. Talbot: **The Complexity of Model Checking Mobile Ambients**. In Foundations of Software Science and Computation Structures (FOSSACS'01), Springer LNCS 2030, 2001, pp 152-167.

[16] W. Charatonik and J.-M. Talbot: **The decidability of model checking mobile ambients.** In Proceedings of the 15th Annual Conference of the European Association for Computer Science Logic, LNCS 2142. Springer, 2001. pp 339-354.

[17] S. Dal-Zilio: **Spatial Congruence for Ambients is Decidable**. Technical Report MSR-TR-2000-41, Microsoft Research, May 2000. Shorter version appears in Proc. of ASIAN'00 - 6th Asian Computing Science Conference, Springer LNCS 1961. 2000.

[18] S. Dal Zilio: **Fixed Points in the Ambient Logic.** In Proc. of FICS 2001 - 3rd Workshop on Fixed Points in Computer Science, September 2001.

[19] M. Dam: **Relevance Logic and Concurrent Composition.** Proceedings LICS'88, IEEE Computer Society, 1988, pp 178-185.

[20] M. Dam: **Proof Systems for π-calculus Logic**. To appear in Logic for Concurrency and Synchronization, Studies in Logic and Computation.

[21] U.H. Engberg, G. Winskel: **Linear Logic on Petri Nets**. BRICS Report RS-94-3, 1994.

[22] J. Engelfriet, Tj. Gelsema: **Multiset and Structural Congruence of the π-calculus with Replication**. TCS (**211**)311-337, 1999.

[23] M.J. Gabbay, A.M. Pitts, **A New Approach to Abstract Syntax Involving Binders**. In Proceedings 14th Annual IEEE Symposium on Logic in Computer Science, Trento, Italy, July 1999. IEEE Computer Society Press, 1999. pp 214-224.

[24] J-Y. Girard, Y. Lafont: **Linear Logic and Lazy Computation**. In Proceedings TAPSOFT 87

(Pisa), LNCS 250 vol. 2, Springer, 1987. pp 53-66.

[25] J.-Y.Girard, Y. Lafont, P. Taylor: **Proofs and Types**. Cambridge University Press, 1989.

[26] A.D. Gordon, L. Cardelli: **Equational Properties of Mobile Ambients**. Mathematical Structures in Computer Science 13(2003):371-408.

[27] M. Hennessy, R. Milner: **Algebraic Laws for Nondeterminism and Concurrency**. JACM, **32**(1)137-161, 1985.

[28] Y. Lafont: **The Linear Abstract Machine**. Theoretical Computer Science (**59**)157-180, 1988.

[29] R. Milner: **Flowgraphs and Flow Algebras**. Journal of the ACM **26**(4), 1979.

[30] R. Milner: **Communicating and Mobile Systems: the π-Calculus**. Cambridge University Press, 1999.

[31] P.W. O'Hearn, D. Pym: **The Logic of Bunched Implications**. Bulletin of Symbolic Logic 5(2), 215-244, 1999.

[32] D. Sangiorgi, **Extensionality and Intensionality in the Ambient Logics**. In Principles of Programming Languages (POPL'01), 2001, pp. 4-13.

[33] D.J. Scott, A. Mycroft, and A.R. Beresford: **Spatial Security Policies for Mobile Agents in a Sentient Computing Environment**. M. Peggé (Ed.): Proc. FASE'03, Springer-Verlag LNCS vol. 2620, April 2003. pp 102-117.

[34] A. Urquhart: **Semantics for Relevant Logics.** Journal of Symbolic Logic **37**(1)159-169, 1972.