Decidability of Modular Logics for Concurrency

Radu Mardare*

Aalborg University, Denmark

Abstract. The modular logics we approach in this paper are logics for concurrent systems that reflect both behavioral and structural properties of systems. They combine dynamic modalities that express behavioural properties of a system with polyadic modalities that join properties of subsystems. Spatial and Separation Logics are examples of modular logics. Being the complex algebraic-coalgebraic semantics, these logics have interesting decidability properties. In this paper we provide a taxonomy of the decision problems for a class of modular logics with semantics based on CCS processes.

1 Introduction

The success of Process Algebras [2] in modelling a wide class of concurrent and distributed systems from Computer Science and Artificial Intelligence to Systems Biology and Biochemistry, raises the necessity to develop analysis techniques for studying and predicting the behaviour of modelled systems. This is the origin of the idea of defining complex query languages specifically designed to express temporal and structural properties of the systems. The dual nature of these calculi—algebraic/equational syntax versus coalgebraic operational semantics, makes them particularly appropriate for a modal logic-based approach.

The process semantics for modal logics are special cases of Kripke semantics: they involve structuring the classes of processes as Kripke models with accessibility relations based on the syntax and the semantics of processes. On one hand, the accessibility relations induced by transition systems have been considered with Hennessy-Milner logic [14] and temporal logics [24]. In addition, mobile, concurrent [12, 22] and dynamic-epistemic [16, 19, 15] features have been added to express more complex semantics. On the other hand, the *spatial logics* [7, 3] use accessibility relations that reflect syntactic properties of processes. They are *intensional logics* [23] able to differentiate bisimilar processes with different structures by using *modular operators* – the logical counterparts of the program constructors of process calculi. Thus, the *parallel operator* specifies properties of complementary (parallel) modules of a program; its adjoint, the *guarantee operator*, quantifies on possible interactive contexts of a program. Some spatial logics consider also operators for specifying the space of computation, such as ambient logic [7], or operators for name passing and name restrictions in process calculi [3], but these are outside the scope of this paper.

* Research supported by Sapere Aude: DFF-Young Researchers Grant 10-085054 of the Danish Council for Independent Research.

By combining the dynamic and the modular operators, one gets an interesting polyadic modal logic. The parallel operator, for instance, is a modal operator of arity 2 that satisfies the axioms of associativity, commutativity, and modal distribution [20, 17, 18]. Similar operators have been studied, e.g., in the context of *Arrow Logic* [1, 13], of *Relevant* and *Substructural Logics* [25], of linear and intensional logics – see [7] for a detailed discussion.

In spite of the similarities with other logics, the combination of dynamic and modular operators raises genuinely new problems concerning decidability and complexity for satisfiability, validity, and model checking against the semantics based on process algebras. In this paper we study and classify by decidability the modular logics for a fragment of CCS [21]. In spite of the restrictive semantics, these logics are already showing interesting behaviours. In this paper we survey the results of [4, 11] (the cases PML_4 and SOL_4) and complete them with new results which improve the state of art and allow us to organize the taxonomy presented in Table 1. We also present some proof methods for process logics that can be further used in various contexts.

Name	**Signature**	**Model checking**	**Satisfiability**
PML_1	$\phi := 0, 1 \mid \neg\phi \mid \phi \wedge \phi \mid \phi \mid \phi \mid \ \diamond\phi$	decidable	unknown
PML_2	$\phi := 0, 1 \mid \neg\phi \mid \phi \wedge \phi \mid \phi \mid \phi \mid \ \langle\alpha\rangle\phi \ \mid \phi \triangleright \phi$	decidable	decidable
PML_3	$\phi := 0, 1 \mid \neg\phi \mid \phi \wedge \phi \mid \phi \mid \phi \mid \langle\alpha, \overline{\alpha}\rangle\phi \mid \phi \triangleright \phi$	decidable	decidable
PML_4	$\phi := 0, 1 \mid \neg\phi \mid \phi \wedge \phi \mid \phi \mid \phi \mid \ \diamond\phi \ \mid \phi \triangleright \phi$	undecidable	undecidable
SOL_0	$\phi := 0, 1 \mid \neg\phi \mid \phi \wedge \phi \mid \quad \exists x.\phi \mid \langle x\rangle\phi$	decidable	unknown
SOL_1	$\phi := 0, 1 \mid \neg\phi \mid \phi \wedge \phi \mid \top \mid \phi \mid \exists x.\phi \mid \langle x\rangle\phi$	decidable	undecidable
SOL_2	$\phi := 0, 1 \mid \neg\phi \mid \phi \wedge \phi \mid \phi \mid \phi \mid \exists x.\phi \mid \langle x\rangle\phi$	decidable	undecidable
SOL_3	$\phi := 0, 1 \mid \neg\phi \mid \phi \wedge \phi \mid \phi \mid \phi \mid \exists x.\phi \mid \langle x\rangle\phi \mid \top \triangleright \phi$	undecidable	undecidable
SOL_4	$\phi := 0, 1 \mid \neg\phi \mid \phi \wedge \phi \mid \phi \mid \phi \mid \exists x.\phi \mid \langle x\rangle\phi \mid \langle\overline{x}\rangle\phi \mid \ \diamond\phi \mid \phi \triangleright \phi$	undecidable	undecidable

Table 1. The decidability problems for modular logics

In [4] it is proved that validity/satisfiability and model checking are undecidable for the logic combining the modular operators with a modality $\diamond$ that encodes the τ-transitions (PML_4) and with second order modalities of type $\exists x.\langle x\rangle\phi$ (SOL_4). We improve these results by showing that the undecidability of satisfiability for modular logics with second order quantifiers derives from the undecidability of a more basic logic that contains second order quantifiers but does not contain the guarantee or the parallel operators. The expressive power of $\top \mid \phi$ is sufficient to generate, in this context, undecidability for satisfiability ($\top$ is "true"). Moreover, the model-checking problem remains undecidable for any similar logic that can express at least $\top \triangleright \phi$ (which is less expressive than the guarantee operator). On the other hand, we prove that the absence of guarantee operator makes the model checking decidable. For the logics without second order operators, we prove that by replacing $\diamond$ (studied in [4]) with a class of operators of type $\langle\alpha, \overline{\alpha}\rangle$, that expresses a synchronization of the action α and its co-action, we obtain a decidable logic even in the presence of parallel and guarantee operators. The same result is obtained by replacing $\diamond$ with the class of dynamic operators $\langle\alpha\rangle$ which

encode atomic actions. We also show that the model-checking problem is decidable for the logic combining parallel and $\diamond$ operators, in the absence of guarantee operator.

2 Preliminaries on Process Algebra

In this section we recall basic notions of process algebra and establish the terminology and the notations. We introduce a finite fragment of CCS [21] which will be used later as the semantics for modular logics. In spite of its simplicity, this fragment is already sufficient to rise important decidability problems for the modular logics.

Definition 1 (CCS processes). *Let Σ be a countable set of* actions *and $0 \notin \Sigma$ a constant called* null process*. The class $\mathbb{P}$ of* CCS processes *is introduced, for arbitrary $\alpha \in \Sigma$, by*

$$P := 0 \mid \alpha.P \mid P|P.$$

Definition 2 (Structural congruence). *The structural congruence is the smallest equivalence relation on $\mathbb{P}$ such that*

1. $(P|Q)|R \equiv P|(Q|R)$ 2. $P|0 \equiv P$ 3. $P|Q \equiv Q|P$
4. *If $P \equiv P'$, then for any $\alpha \in \Sigma$ and $Q \in \mathbb{P}$, $\alpha.P \equiv \alpha.P'$ and $P|Q \equiv P'|Q$.*

Definition 3 (Operational semantics). *Let $\tau \notin \Sigma \cup \mathbb{P}$ and consider an involution on Σ that associates to each action $\alpha \in \Sigma$ its co-action $\overline{\alpha}$, such that $\overline{\overline{\alpha}} = \alpha$ and $\overline{\alpha} \neq \alpha$. The operational semantics presented bellow defines a labeled transition system $\mathbb{T} : \mathbb{P} \to (\Sigma \cup \{\tau\}) \times \mathbb{P}$, where $\mathbb{T}(P) = (\mu, Q)$ is denoted by $P \xrightarrow{\mu} Q$ for any $\mu \in \Sigma \cup \{\tau\}$.*

$$\frac{}{\alpha.P \xrightarrow{\alpha} P}, \alpha \in \Sigma \qquad \frac{}{\alpha.P|\overline{\alpha}.Q \xrightarrow{\tau} P|Q}, \alpha \in \Sigma$$

$$\frac{P \equiv Q \qquad P \xrightarrow{\mu} P'}{Q \xrightarrow{\mu} P'}, \mu \in \Sigma \cup \{\tau\} \qquad \frac{P \xrightarrow{\mu} P'}{P|Q \xrightarrow{\mu} P'|Q}, \mu \in \Sigma \cup \{\tau\}$$

In this paper we consider, in addition, the transitions labeled by pairs of complementary actions $(\alpha, \overline{\alpha})$ defined by $\alpha.P'|\overline{\alpha}.P''|P''' \xrightarrow{\alpha,\overline{\alpha}} P'|P''|P'''$. We call a process P *guarded* if $P \equiv \alpha.Q$ for some $\alpha \in \Sigma$ and we use the notation $P^k \stackrel{def}{=} \underbrace{P|...|P}_{k}$ for $k \geq 1$.

Definition 4. *For an arbitrary process $P \in \mathbb{P}$, let $Act(P) \subset \Sigma$ be defined by*

1. $Act(0) \stackrel{def}{=} \emptyset$ 2. $Act(\alpha.P) \stackrel{def}{=} \{\alpha\} \cup Act(P)$ 3. $Act(P|Q) \stackrel{def}{=} Act(P) \cup Act(Q)$.

For $\Omega \subseteq \Sigma$ and h, w nonnegative integers we define the class $\mathbb{P}^{\Omega}_{(h,w)}$ of processes with actions from Ω and syntactic trees bound by two dimensions: the *depth h* of the tree and the *width w* (this represents the maximum number of structural congruent processes that can be found in a node of the tree). $\mathbb{P}^{\Omega}_{(h,w)}$ is introduced inductively on h.

$\mathbb{P}^{\Omega}_{(0,w)} = \{0\}$;

$\mathbb{P}^{\Omega}_{(h+1,w)} = \{(\alpha_1.P_1)^{k_1}|...|(\alpha_i.P_i)^{k_i},$ for $k_j \leq w, \alpha_j \in \Omega, P_j \in \mathbb{P}^{\Omega}_{(h,w)}, \forall j = 1..i\}$.

Observe that if $\Omega \subseteq \Sigma$ is a finite set, then $\mathbb{P}^{\Omega}_{(h,w)}$ is a finite set of processes.

In what follows, we introduce *structural bisimulation*, a relation on processes similar to the *pruning relation* proposed for trees (static ambient processes) in [5]. This relation will play an essential role in establishing the bounded model property for some modular logics. The structural bisimulation is indexed by a set $\Omega \subseteq \Sigma$ of actions and by two nonnegative integers h, w. Intuitively, two processes are *Ω-structural bisimilar* on size (h, w) if they look indistinguishable for an external observer that sees only the actions in Ω, does not follow a process for more than h transition steps, and cannot distinguish more than w cloned parallel subprocesses of an observed process.

Definition 5 (Ω-Structural Bisimulation). *Let $\Omega \subseteq \Sigma$ and h, w two nonnegative integers. The Ω-structural bisimulation on $\mathbb{P}$, $\approx^{\Omega}_{(h,w)}$, is defined inductively as follows.*
If $P \equiv Q \equiv 0$, then $P \approx^{\Omega}_{(h,w)} Q$;
If $P \not\equiv 0$ and $Q \not\equiv 0$, then
$P \approx^{\Omega}_{(0,w)} Q$ *always.*
$P \approx^{\Omega}_{(h+1,w)} Q$ *iff for any $i \in 1..w$ and any $\alpha \in \Omega$:*

- $P \equiv \alpha.P_1|...|\alpha.P_i|P'$ *implies* $Q \equiv \alpha.Q_1|...|\alpha.Q_i|Q'$, $P_j \approx^{\Omega}_{(h,w)} Q_j$, $j = 1..i$;
- $Q \equiv \alpha.Q_1|...|\alpha.Q_i|Q'$ *implies* $P \equiv \alpha.P_1|...|\alpha.P_i|P'$, $Q_j \approx^{\Omega}_{(h,w)} P_j$, $j = 1..i$.

We emphasize further some properties of Ω-structural bisimulation. The proofs of these results can be found in Appendix.

Lemma 1 (Equivalence). *$\approx^{\Omega}_{(h,w)}$ is an equivalence relations on $\mathbb{P}$.*

Lemma 2 (Congruence). *Let $\Omega \subseteq \Sigma$ be a set of actions.*
1. If $P \approx^{\Omega}_{(h,w)} Q$, then $\alpha.P \approx^{\Omega}_{(h+1,w)} \alpha.Q$.
2. If $P \approx^{\Omega}_{(h,w)} P'$ and $Q \approx^{\Omega}_{(h,w)} Q'$, then $P|Q \approx^{\Omega}_{(h,w)} P'|Q'$.

For nonnegative integers h, h', w, w' we convey to write $(h', w') \leq (h, w)$ iff $h' \leq h$ and $w' \leq w$.

Lemma 3 (Restriction). *Let $\Omega' \subseteq \Omega \subseteq \Sigma$ and $(h', w') \leq (h, w)$. If $P \approx^{\Omega}_{(h,w)} Q$, then $P \approx^{\Omega'}_{(h',w')} Q$.*

Lemma 4 (Split). *If $P'|P'' \approx^{\Omega}_{(h,w_1+w_2)} Q$ for some $\Omega \subseteq \Sigma$, then there exist $Q', Q'' \in \mathbb{P}$ such that $Q \equiv Q'|Q''$ and $P' \approx^{\Omega}_{(h,w_1)} Q'$, $P'' \approx^{\Omega}_{(h,w_2)} Q''$.*

Lemma 5 (Step-wise propagation). *If $P \approx^{\Omega}_{(h,w)} Q$ and $P \xrightarrow{\alpha} P'$ for some $\alpha \in \Omega \subseteq \Sigma$, then there exists a transition $Q \xrightarrow{\alpha} Q'$ such that $P' \approx^{\Omega}_{(h-1,w-1)} Q'$; if $P \xrightarrow{\alpha,\overline{\alpha}} P'$, then there exists a transition $Q \xrightarrow{\alpha,\overline{\alpha}} Q'$ such that $P' \approx^{\Omega}_{(h-2,w-2)} Q'$.*

As Σ is a denumerable set, assume a lexicographic order $\ll \subseteq \Sigma \times \Sigma$ on it. Then, any element $\alpha \in \Sigma$ has a successor denoted by $succ(\alpha)$ and any finite subset $\Omega \subset \Sigma$ has a maximum element denoted by $sup(\Omega)$. We define $\Omega^+ = \Omega \cup \{succ(sup(\Omega))\}$.

The next lemma states that for any finite set Ω and any nonnegative integers h, w, the equivalence relation $\approx^{\Omega}_{(h,w)}$ partitions the class $\mathbb{P}$ of processes in equivalence classes such that each equivalence class has a representative in the finite set $\mathbb{P}^{\Omega^+}_{(h,w)}$.

Lemma 6 (Representation Theorem). *For any finite set $\Omega \subseteq \Sigma$, any nonnegative integers h, w and any process $P \in \mathbb{P}$, there exists a process $Q \in \mathbb{P}^{\Omega^+}_{(h,w)}$ such that $P \approx^{\Omega}_{(h,w)} Q$.*

3 Modular Logic

In this section we introduce the modular logics. One class contains the *propositional modular logics* (PMLs) that extend the classic propositional logic with modular and dynamic operators. The other class consists of *second order modular logics* (SOLs) that are equipped with variables and quantifiers over modalities.

Definition 6 (Syntax). *Let Σ and X be two disjoint countable sets. Consider the logics defined for arbitrary $\alpha \in \Sigma$ and $x, y \in X$ as follows.*

$$
\begin{array}{ll}
PML_1 & \phi := 0, 1 \mid \neg\phi \mid \phi \wedge \phi \mid \phi|\phi \mid \quad \diamond \phi \\
PML_2 & \phi := 0, 1 \mid \neg\phi \mid \phi \wedge \phi \mid \phi|\phi \mid \langle\alpha\rangle\phi \quad \mid \phi \triangleright \phi \\
PML_3 & \phi := 0, 1 \mid \neg\phi \mid \phi \wedge \phi \mid \phi|\phi \mid \langle\alpha, \overline{\alpha}\rangle\phi \mid \phi \triangleright \phi \\
PML_4 & \phi := 0, 1 \mid \neg\phi \mid \phi \wedge \phi \mid \phi|\phi \mid \quad \diamond \phi \quad \mid \phi \triangleright \phi
\end{array}
$$

$$
\begin{array}{ll}
SOL_0 & \phi := 0, 1 \mid \neg\phi \mid \phi \wedge \phi \mid \qquad \exists x.\phi \mid \langle x\rangle\phi \\
SOL_1 & \phi := 0, 1 \mid \neg\phi \mid \phi \wedge \phi \mid \top|\phi \mid \exists x.\phi \mid \langle x\rangle\phi \\
SOL_2 & \phi := 0, 1 \mid \neg\phi \mid \phi \wedge \phi \mid \phi|\phi \mid \exists x.\phi \mid \langle x\rangle\phi \\
SOL_3 & \phi := 0, 1 \mid \neg\phi \mid \phi \wedge \phi \mid \phi|\phi \mid \exists x.\phi \mid \langle x\rangle\phi \mid \top \triangleright \phi \\
SOL_4 & \phi := 0, 1 \mid \neg\phi \mid \phi \wedge \phi \mid \phi|\phi \mid \exists x.\phi \mid \langle x\rangle\phi \mid \langle\overline{x}\rangle\phi \mid \ \diamond \phi \mid \phi \triangleright \phi
\end{array}
$$

Here 0 and 1 are modal operators of arity 0 that characterize the null process and the guarded processes respectively.

The semantics is given for the class $\mathbb{P}$ of CCS processes as frames. In particular, a definition of the satisfiability operator, $P \models \phi$ that relates a process $P \in \mathbb{P}$ with the property ϕ written in the syntax of *PMLs*, is given.

Definition 7 (Semantics of PMLs). *Let $P \in \mathbb{P}$ and ϕ a formula of PML_i, $i = 1..4$. The relation $P \models \phi$ is defined inductively as follows.*

$P \models 0$ *iff* $P \equiv 0$.

$P \models 1$ *iff there exist* $\alpha \in \Sigma$ *and* $Q \in \mathbb{P}$ *such that* $P \equiv \alpha.Q$.

$P \models \neg\phi$ *iff* $P \not\models \phi$.

$P \models \phi \wedge \psi$ *iff* $P \models \phi$ *and* $P \models \psi$.

$P \models \phi|\psi$ *iff* $P \equiv Q|R$, $Q \models \phi$ *and* $R \models \psi$.

$P \models \top|\phi$ *iff* $P \equiv Q|R$ *and* $R \models \phi$.

$P \models \diamond\phi$ *iff there exists a transition* $P \xrightarrow{\tau} P'$ *and* $P' \models \phi$.

$P \models \langle\alpha\rangle\phi$ *iff there exists a transition* $P \xrightarrow{\alpha} P'$ *and* $P' \models \phi$.

$P \models \langle\alpha, \overline{\alpha}\rangle\phi$ *iff there exists a transition* $P \xrightarrow{\alpha,\overline{\alpha}} P'$ *and* $P' \models \phi$.

$P \models \phi \triangleright \psi$ *iff for any* $Q \in \mathbb{P}$, $Q \models \phi$ *implies* $P|Q \models \psi$.

$P \models \top \triangleright \phi$ *iff for any* Q, $P|Q \models \phi$.

Observe that, equivalently, we can introduce the semantics in the modal logic fashion by defining a frame for PMLs as the structure

$\mathcal{M} = (\mathbb{P}, i, \mathcal{R}_0, \mathcal{R}_1, (\mathcal{R}_\alpha)_{\alpha\in\Sigma}, (\mathcal{R}_{(\alpha,\overline{\alpha})})_{\alpha\in\Sigma}, \mathcal{R}_\tau, \mathcal{R}_|, \mathcal{R}_\triangleright)$ where
$i : \mathbb{P} \to 2^{\{0\}}$ is the interpretation function defined by $i(P) = \{0\}$ for $P \equiv 0$ and $i(P) = \emptyset$ else;

$\mathcal{R}_0 = \{0\}$ and $\mathcal{R}_1 = \{\alpha.P : \alpha \in \Sigma, P \in \mathbb{P}\}$ are accesibility relations of arity 1;
$(\mathcal{R}_\alpha)_{\alpha\in\Sigma}$ is a class of accessibility relations $\mathcal{R}_\alpha \subseteq \mathbb{P} \times \mathbb{P}$ indexed by actions and defined by $(P, Q) \in \mathcal{R}_\alpha$ iff $P \xrightarrow{\alpha} Q$.
$(\mathcal{R}_{(\alpha,\overline{\alpha})})_{\alpha\in\Sigma}$ is a class of accessibility relations indexed by pairs of complementary actions and defined by $(P, Q) \in \mathcal{R}_{(\alpha,\overline{\alpha})}$ iff $P \xrightarrow{\alpha,\overline{\alpha}} Q$.
$\mathcal{R}_\tau$ is an accessibility relations $\mathcal{R}_\tau$ defined by $(P, Q) \in \mathcal{R}_\alpha$ iff $P \xrightarrow{\tau} Q$.
$\mathcal{R}_| \subseteq \mathbb{P} \times \mathbb{P} \times \mathbb{P}$ is a relation defined by $(P, Q, R) \in \mathcal{R}_|$ iff $P \equiv Q|R$
$\mathcal{R}_\triangleright \subseteq \mathbb{P} \times \mathbb{P} \times \mathbb{P}$ is a relation defined by $(P, Q, R) \in \mathcal{R}_\triangleright$ iff $R \equiv P|Q$.
In this presentation $0, 1$ are modal operators of arity 0, $\langle\alpha\rangle$, $\langle\alpha, \overline{\alpha}\rangle$ and $\diamond$ are modal operators of arity 1, while $|$ and $\triangleright$ are modal operators of arity 2.

Before introducing the semantics of second order modular logics (SOLs), we should stress the fact that in our syntax X is a set of variables that will be interpreted over Σ. As usual, we call an occurrence of a variable $x \in X$ in a formula ϕ (written in the syntax of SOL_i, $i = 0, \ldots, 4$) a *free occurrence* if it is not in the scope of a quantifier $\exists x$. We call a variable x a *free variable* in a formula if it has at least one free occurrence[1]. A formula ϕ is *closed* if it contains no free variables; else, we call it *open*. A valuation $v : X \hookrightarrow \Sigma$ is a partial function that associates values in Σ to some variables in X. If v is a valuation, $x \in X$ is a variable that is not in the domain of v, and $\alpha \in \Sigma$, we denote by $v_{\{x\to\alpha\}}$ the valuation v' that extends v with $v'(x) = \alpha$.

The semantics of second order modular logics (SOLs) is given by the satisfiability operator, $P, v \models \phi$ that relates a process $P \in \mathbb{P}$ and valuation $v : X \to \Sigma$ interpreting the free variable of ϕ, to a well formed formula ϕ of SOL_i, $i = 0, \ldots, 4$.

Definition 8 (Semantics of SOLs). *The relation $P, v \models \phi$ is defined as follows.*

$P, v \models 0$ *iff* $P \equiv 0$.
$P, v \models 1$ *iff there exists* $\alpha.Q \in \mathbb{P}$ *such that* $P \equiv \alpha.Q$.
$P, v \models \neg\phi$ *iff* $P, v \not\models \phi$.
$P, v \models \phi \wedge \psi$ *iff* $P, v \models \phi$ *and* $P, v \models \psi$.
$P, v \models \phi|\psi$ *iff* $P \equiv Q|R$, $Q, v \models \phi$ *and* $R, v \models \psi$.
$P, v \models \top|\phi$ *iff* $P \equiv Q|R$, $Q, v \models \phi$.
$P, v \models \diamond\phi$ *iff* $P \xrightarrow{\tau} P'$ *and* $P', v \models \phi$.
$P, v \models \langle x\rangle\phi$ *iff* $P \xrightarrow{v(x)} P'$ *and* $P', v \models \phi$.
$P, v \models \langle \overline{x}\rangle\phi$ *iff* $P \xrightarrow{\overline{v(x)}} P'$ *and* $P', v \models \phi$.
$P, v \models \phi \triangleright \psi$ *iff for any process* $P' \in \mathbb{P}$, $P', v \models \phi$ *implies* $P'|P, v \models \psi$.
$P, v \models \top \triangleright \phi$ *iff for any process* P', $P'|P, v \models \phi$.
$P, v \models \exists x.\phi$ *iff there exists* $\alpha \in \Sigma$ *such that* $P, v_{\{\alpha\to x\}} \models \phi$.

In addition to the boolean operators we also introduce the next derived operators that will be used both with PMLs and SOLs.

$$\top \stackrel{def}{=} 0 \vee \neg 0 \qquad \bot \stackrel{def}{=} \neg\top \qquad \phi \parallel \psi \stackrel{def}{=} \neg(\neg\phi|\neg\psi)$$
$$\circ\phi \stackrel{def}{=} (\neg\phi) \triangleright \bot \qquad \phi^\forall \stackrel{def}{=} \phi \parallel \top \qquad \alpha.\phi \stackrel{def}{=} 1 \wedge \langle\alpha\rangle\phi$$

[1] As usual, we assume that variables occurring under different boundaries or both bound and free do not clash, even if the same (meta-)symbol $x \in X$ is used to name them.

$\top$ and $\bot$ are boolean constants, hence $\top|\phi$ and $\top \triangleright \phi$ are particular instances of $\psi|\phi$ and $\psi \triangleright \phi$ respectively. Notice that in the logics where $\phi|\psi$ is a legal construction, 1 can be defined from 0 by $1 \stackrel{def}{=} \neg 0 \wedge (0 \parallel 0)$. Observe also that the operator $\circ$, that can be defined in a logic where $\phi \triangleright \psi$ is legal, is a universal modality and $\circ\phi$ encodes the validity of ϕ over $\mathbb{P}$.

Definition 9. *A formula ϕ of PMLs is* satisfiable *if there exists a process $P \in \mathbb{P}$ such that $P \models \phi$; it is valid (a* validity*) if for any process $P \in \mathbb{P}$, $P \models \phi$. A closed formula ϕ of SOLs is* satisfiable *if there exists a process $P \in \mathbb{P}$ such that $P, \emptyset \models \phi$, where $\emptyset$ is the empty valuation; it is valid (a* validity*) if for any process $P \in \mathbb{P}$ $P, \emptyset \models \phi$.*

We denote the fact that ϕ is a validity by $\models \phi$. Hereafter, we call the **satisfiability problem** (**validity problem**) for a logic against a given semantics the problem of deciding if an arbitrary formula is satisfiable (valid). The **model checking problem** for PMLs consists in deciding, for an arbitrary formula ϕ and an arbitrary process P, if $P \models \phi$. The same problem for SOLs consists in deciding, for an arbitrary closed formula ϕ and an arbitrary process P, if $P, \emptyset \models \phi$.

Observe that Definition 9 implies that ϕ is a validity iff $\neg\phi$ is not satisfiable and reverse, ϕ is satisfiable iff $\neg\phi$ is not valid. Consequently, satisfiability and validity are dual problems implying that once one has been proved decidable/undecidable, the other shares the same property.

4 Decision problems for Second Order Modular Logics

In [4] it is proved that SOL_4 is undecidable. The proof is based on the method proposed previously in [11] where it is shown that the second order quantifiers (over ambient names) in ambient logic, in combination with the parallel operator, can induce undecidability for satisfiability. A corollary of this result is the undecidability of SOL_2. In what follows, we use the same method for proving a stronger result, i.e. that satisfiability for SOL_1 is undecidable. This result shows that even in absence of the parallel operator (in SOL_1, the parallel operator can only appear in constructions of type $\top|\phi$) second order quantification implies the undecidability of satisfiability for SOL_2, SOL_3 and SOL_4.

In the second part of this section we approach the model checking problem. For SOL_2 model checking is decidable (implying decidability of model checking for both SOL_1 and SOL_0), while for SOL_3 model checking is undecidable (implying the undecidability of model checking for SOL_4). This shows that the expressivity of guarantee operator is not the one responsable of the undecidability of model checking, but the expressivity of $\top \triangleright \phi$. Notice that $P, v \models \top \triangleright \phi$ implies that all processes having P as subprocess have the property ϕ under the evaluation v, i.e. we face a universal quantification over the class of upper processes of P. The satisfiability of SOL_0 is open.

4.1 The satisfiability problem

In what follows, we prove that the satisfiability problem of SOL_1 is equivalent with the satisfiability problem of a fragment of first order logic known to be undecidable

for finite domains[2]. This fragment is FOL introduced inductively, for a single binary predicate $p(x, y)$ and for $x, y \in X$, by:

$$f := p(x, y) \mid \neg f \mid f \wedge f \mid \exists x.f.$$

The semantics of FOL is defined for a finite domain $D \subseteq \Sigma$, for an interpretation $I \subseteq D \times D$ of the predicate and for a valuation $v : X \to D$ as follows.

$(D, I), v \models p(x, y)$ iff $(v(x), v(y)) \in I$

$(D, I), v \models \neg f$ iff $(D, I), v \not\models f$

$(D, I), v \models f \wedge g$ iff $(D, I), v \models f$ and $(D, I), v \models g$

$(D, I), v \models \exists x.f$ iff there exists $\alpha \in D$ and $(D, I), v_{\{x \to \alpha\}} \models f$.

It is known that satisfiability for FOL is undecidable. We will prove further that satisfiability of FOL is equivalent with satisfiability for SOL_1.

We begin by describing a special class $\mathcal{P} \subseteq \mathbb{P}$ of processes that can be characterized by the formulas of SOL_1.

Consider the following derived operators in SOL_1:
$D(x) = \langle x \rangle 0$, $R(x, y) = 1 \wedge \langle x \rangle \langle y \rangle 0$ and
$Model = [(1 \to (\exists x D(x) \vee \exists x \exists y R(x, y)))|\top] \wedge [\forall x \forall y((R(x, y)|\top) \to (D(x)|\top \wedge D(y)|\top))]$.

We prove that $Model$ characterizes the class $\mathcal{P}$ of process containing 0 and all processes of type $\alpha_1.0|...|\alpha_k.0 \mid \alpha_{i_1}.\alpha_{j_1}.0|...|\alpha_{i_l}.\alpha_{j_l}.0$ for $i_1, ..., i_l, j_1, .., j_l \in \{1, ..k\}$.

Lemma 7. *$P, v \models Model$ iff either $P \in \mathcal{P}$.*

Proof. $P, v \models (1 \to (\exists x D(x) \vee \exists x \exists y R(x, y)))|\top$ iff for any Q s.t. $P \equiv Q|R$ we have that if $Q \models 1$ (i.e. $Q \equiv \alpha.Q'$ for some α), then $Q' \equiv 0$ or $Q' \equiv \beta.0$. Hence, $Q \models 1$ implies $Q \equiv \alpha.0$ or $Q \equiv \alpha.\beta.0$ for some $\alpha, \beta \in \Sigma$. Moreover, $P, v \models \forall x \forall y((R(x, y)|\top) \to (D(x)|\top \wedge D(y)|\top))$ iff $P \equiv \alpha.\beta.0|Q$ implies $P \equiv \alpha.0|\beta.0|P'$.

Now, we describe a method for associating to each pair (D, I) used in the semantics of FOL, a process $P^I_D \in \mathcal{P}$.

Let $D \subseteq \Sigma$ be a finite set and $I \subset D \times D$ a relation on D. Suppose that $D = \{\alpha_1, ...\alpha_k\}$ with $k \geq 1$, and $I = \{(\alpha_{i_1}, \alpha_{j_1}), (\alpha_{i_2}, \alpha_{j_2}), ..., (\alpha_{i_l}, \alpha_{j_l})\}$, with $i_1, ..., i_l, j_1, .., j_l \in \{1, ..k\}$. We denote by $Dom(\Sigma)$ the class of these pairs (D, I). We associate to each pair $(D, I) \in Dom(\Sigma)$ the process $P^I_D \in \mathcal{P}$ defined by $P^I_D \equiv \alpha_1.0|...|\alpha_k.0 \mid \alpha_{i_1}.\alpha_{j_1}.0|...|\alpha_{i_l}.\alpha_{j_l}.0$.
Reverse, consider a process $P \in \mathcal{P}$ for which there exists $\alpha_1, ...\alpha_k \in \Sigma$, not necessarily distinct, and $i_1, ..., i_l, j_1, .., j_l \in \{1, ..k\}$ s.t. $P \equiv \alpha_1.0|...|\alpha_k.0 \mid \alpha_{i_1}.\alpha_{j_1}.0|...|\alpha_{i_l}.\alpha_{j_l}.0$.
We take $D = \{\alpha_1, ...\alpha_k\}$ and $I = \{(\alpha_{i_1}, \alpha_{j_1}), (\alpha_{i_2}, \alpha_{j_2}), ..., (\alpha_{i_l}.\alpha_{j_l})\}$ and this is the pair we associate to P. Notice that, by construction, if $\alpha_i = \alpha_j$ then it appears in D only once and similarly, if $(\alpha_{i_s}, \alpha_{j_s}) = (\alpha_{i_t}, \alpha_{j_t})$ for some $s \neq t$, then it is taken only once in I.

For proving the equivalence between the two decidability problems, we define the encoding [ ] that associates each formula of FOL to a formula of SOL_1 by
$[p(x, y)] = R(x, y)|\top$; $[\neg f] = \neg[f]$; $[f \wedge g] = [f] \wedge [g]$; $[\exists x.f] = \exists x.((D(x)|\top) \wedge [f])$.

Lemma 8. *$(D, I), v \models f$ iff $P^D_I, v \models [f]$.*

[2] The same fragment of first order logic is used in [4] for proving the undecidability of SOL_4.

Proof. We prove it by induction on $f \in FOL$. The non-trivial cases are:

The case $f = \exists x.g$**:** $(D, I), v \models \exists x.g$ iff there exists $\alpha \in D$ s.t. $(D, I), v_{\{x\to\alpha\}} \models g$. Further, the inductive hypothesis gives $P_I^D, v_{\{x\to\alpha\}} \models [g]$. But because $P_I^D \equiv \alpha.0|P'$, we obtain that $P_I^D, v_{\{x\to\alpha\}} \models D(x)|\top$. Hence $P_I^D, v_{\{x\to\alpha\}} \models D(x)|\top \wedge [g]$ that implies $P_I^D, v \models \exists x.(D(x)|\top \wedge [g])$ that is equivalent with $P_I^D, v \models [f]$.

The case $f = p(x, y)$**:** $(D, I), v \models p(x, y)$ iff $(v(x), v(y)) \in I$. But this is equivalent with $P_I^D \equiv v(x).v(y).0|P'$ that implies $P_I^D, v \models (1 \wedge \langle x\rangle\langle y\rangle.0)|\top$, i.e. $P_I^D, v \models [f]$.

Lemma 9. *Let f be a closed formula of FOL. Then f is satisfiable in FOL iff $Model \wedge [f]$ is satisfiable in SOL_1.*

Proof. *Model* characterizes the class $\mathcal{P}$ of processes. So, if there exists a model $(D, I) \in Dom(\Sigma)$ such that $(D, I), \emptyset \models f$, then $P_I^D, \emptyset \models Model \wedge [f]$, where $\emptyset$ is the empty valuation. Reverse, if there is a process $P \in \mathbb{P}$ that satisfies $Model \wedge [f]$, then $P, \emptyset \models Model$, i.e. $P \in \mathcal{P}$ meaning that there exists $(D, I) \in Dom(\Sigma)$ such that $P \equiv P_I^D$. Then $P_I^D, \emptyset \models [f]$ that implies $(D, I), \emptyset \models f$.

Theorem 1. *For SOL_1 validity and satisfiability are undecidable.*

This result implies the undecidability of satisfiability for the more expressive logics.

Corollary 1. *The satisfiability is undecidable for SOL_2 and SOL_3.*

4.2 The model-checking problem

With model checking the situation is different. The simple presence of second order quantification does not imply undecidability, as for the case of satisfiability.

Theorem 2. *For SOL_2 model checking is decidable.*

Proof. Observe that for arbitrary P, ϕ, v' and $\alpha, \beta \notin Act(P) \cup dom(v')$, we have $P, v'_{\{x\to\alpha\}} \models \phi$ iff $P, v'_{\{x\to\beta\}} \models \phi$. Due to this property, for deciding $P, v \models \phi$ it is sufficient to only consider the valuations assigning values from $Act(P) \cup \{\alpha_1, ..., \alpha_k\}$ to the free variables of ϕ, where $\alpha_i \notin Act(P)$ for $i = 1..k$ and k is the number of distinct variables that appear in ϕ. Further, one can prove that the operators can be eliminated inductively, and the model-checking problem can be reduced, at each step, to a finite number of model-checking problems involving the subprocesses of P (which are finitely many, modulo structural congruence) and the subformulas of ϕ.

Corollary 2. *The model-checking problems for SOL_1 and SOL_0 are decidable.*

The presence in a logic of the operator $\top \triangleright \phi$ is sufficient to make the model-checking problem undecidable. Notice that $P, v \models \top \triangleright \phi$ involves a universal quantification over the class of upper processes of P.

Theorem 3. *For SOL_3 model checking is undecidable.*

Proof. The proof is based on the observation, emphasized also in [11], that in any logic which can express $\top \triangleright \phi$, the decidability of model-checking problem implies the decidability of satisfiability. Hence, the undecidability of satisfiability implies undecidability of model checking. Indeed, for an arbitrary formula ϕ in SOL_3, it is trivial to verify that $\models \phi$ iff $0, \emptyset \models \top \triangleright \phi$. As for SOL_3 validity is undecidable, we obtain undecidability for model checking.

5 Decision problems for Propositional Modular Logics

In this section we focus on propositional modular logics. In [4] it has been proved that for PML_4 satisfiability, validity and model checking are undecidable against the semantics presented in Section 3. The proof is based on the equivalence between the satisfiability problems for SOL_4 and PML_4. The result reveals that the combination of the modality $\diamond$ based on τ-transition with the modular operators $|$ and $\triangleright$, generates undecidability.

In this section we show that the combination of the two modular operators and a transition-based modality does not always produce undecidability. We will show that for PML_2 and PML_3 both the satisfiability/validity and model-checking problems are decidable. PML_2 contains dynamic operators indexed by actions $\langle\alpha\rangle$ that reflect the interleaving semantics of CCS. PML_3 is closer to PML_4 as it expresses communications by the dynamic operators $\langle\alpha, \overline{\alpha}\rangle$. But while the communications reflected by $\diamond$ have an anonymous status, the communications expressible in PML_3 can also specify the pairs of actions involved. Observe that $\diamond$ can be seen as an existential quantifier over the class of $\langle\alpha, \overline{\alpha}\rangle$ and in this sense PML_4 has a second-order nature that might explain its undecidability.

In the second part of the section we consider the logic PML_1 which combines $\diamond$ with the parallel operator. We prove that for this logic model checking is decidable. However, the satisfiability for PML_1 is an open problem.

5.1 Decidability of PML_2 and PML_3

We prove that for PML_2 and PML_3 satisfiability/validity problems are decidable. The proofs are based on the *bounded model property* technique which consists in showing that, given a formula ϕ of PML_2 or PML_3, we can identify a finite class of processes $\mathbb{P}_\phi$, bound by the dimension of ϕ, such that if ϕ has a model in $\mathbb{P}$, then it has a model in $\mathbb{P}_\phi$. Thus, the satisfiability problem in $\mathbb{P}$ is equivalent with the satisfiability in $\mathbb{P}_\phi$. This result can be further used to prove the decidability of satisfiability for the two logics. Indeed, as $\mathbb{P}_\phi$ is finite, checking the satisfiability of a formula can be done by exhaustively verifying it for all the processes in $\mathbb{P}_\phi$.

The method adapted for modular logics was first proposed in [6] and reused in [5] for the case of static ambient logic. It consists in identifying a structural equivalence on processes, sensitive to the dimension of the logical formulas, that relates two processes whenever they satisfy the same formulas of a given size. In our case this relation is the structural bisimulation defined in Section 2.

Definition 10 (Size of a formula). *The* sizes of a formula *of PML_3, denoted by $(\!|\phi|\!) = (h, w)$, is defined inductively assuming that $(\!|\phi|\!) = (h, w)$ and $(\!|\psi|\!) = (h', w')$, as follows.*

1. $(\!|0|\!) \stackrel{def}{=} (1, 1)$.
2. $(\!|\neg\phi|\!) \stackrel{def}{=} (\!|\phi|\!)$.
3. $(\!|\phi \wedge \psi|\!) \stackrel{def}{=} (max(h, h'), max(w, w'))$.
4. $(\!|\langle\alpha\rangle\phi|\!) \stackrel{def}{=} (h + 1, w + 1)$.
5. $(\!|\phi \triangleright \psi|\!) \stackrel{def}{=} (max(h, h'), w + w')$.
6. $(\!|\phi|\psi|\!) \stackrel{def}{=} (max(h, h'), w + w')$.
7. $(\!|\langle\alpha, \overline{\alpha}\rangle\phi|\!) \stackrel{def}{=} (h + 2, w + 2)$.

Definition 11. *The set of actions of a formula ϕ, $act(\phi) \subseteq \Sigma$ is given by:*

1. $act(0) \stackrel{def}{=} \emptyset$
2. $act(\neg\phi) = act(\phi)$
3. $act(\phi \wedge \psi) \stackrel{def}{=} act(\phi) \cup act(\psi)$
4. $act(\langle\alpha\rangle\phi) \stackrel{def}{=} \{\alpha\} \cup act(\phi)$
5. $act(\phi \triangleright \psi) \stackrel{def}{=} act(\phi) \cup act(\psi)$
6. $act(\phi|\psi) \stackrel{def}{=} act(\phi) \cup act(\psi)$
7. $act(\langle\alpha, \overline{\alpha}\rangle\phi) \stackrel{def}{=} \{\alpha, \overline{\alpha}\} \cup act(\phi)$.

The next Lemma states that a formula ϕ of PML_2 or PML_3 expresses a property of a process P up to $\approx^{act(\phi)}_{(\!|\phi|\!)}$. A sketch of its proof can be found in Appendix.

Lemma 10. *The next assertion is true for PML_2 and PML_3.*
If $P \approx^{act(\phi)}_{(\!|\phi|\!)} Q$, then $[P \models \phi$ iff $Q \models \phi]$.

This result guarantees the bounded model property for both PML_2 and PML_3.

Theorem 4 (Bounded model property). *For PML_2 and PML_3,*
if $P \models \phi$, then there exists $Q \in \mathbb{P}^{act(\phi)^+}_{(\!|\phi|\!)}$ such that $Q \models \phi$.

Theorem 5 (Decidability). *For PML_2 and PML_3 validity and satisfiability are decidable against process semantics.*

Proof. The decidability of satisfiability derives, for both logics, from the bounded model property. Indeed, if ϕ has a model, by Lemma 4, it has a model in $\mathbb{P}^{act(\phi)^+}_{(\!|\phi|\!)}$. As $act(\phi)$ is finite, $\mathbb{P}^{act(\phi)^+}_{(\!|\phi|\!)}$ is finite. Hence, checking for membership is decidable.

5.2 The model-checking problems

We focus now on the model-checking problems. We start by stating the decidability of model checking for PML_2 and PML_3.

Theorem 6. *For PML_2 and PML_3 model checking is decidable.*

Proof. Given the process P and the formula ϕ, we show inductively on the structure of ϕ that $P \models \phi$ is decidable, by showing that the problem can be reduced, step by step, to a finite number of model checking problems involving subformulas of ϕ. The only interesting case is $\phi = \phi_1 \triangleright \phi_2$. Due to the bounded model property, $P \models \phi_1 \triangleright \phi_2$ iff for any $Q \in \mathbb{P}^{act(\phi_1)^+}_{(\!|\phi_1|\!)}$ we have that $Q \models \phi_1$ implies $P|Q \models \phi_2$. As there are only a finite number of processes $Q \in \mathbb{P}^{act(\phi_1)^+}_{(\!|\phi_1|\!)}$, we are done.

Theorem 7. *For PML_1 model checking is decidable.*

Proof. As before, we reduce the problem $P \models \phi$ to a finite number of model checking problems involving subprocesses of P (we do not have $\triangleright$) and subformulas of ϕ. The only difference w.r.t. PML_2 or PML_3 is case $\phi = \diamond\psi$. We have $P \models \diamond\psi$ iff there exists a transition $P \stackrel{\tau}{\longrightarrow} P'$ such that $P' \models \psi$. But the number of processes P' such that $P \stackrel{\tau}{\longrightarrow} P'$ is finite modulo structural congruence. Hence, also in this case, the problem can be reduced to a finite number of model checking problems that refers to ψ.

6 Conclusive remarks

The goal of this paper was to present the taxonomy in Table 1. The results improve the state of the arts: the undecidability of satisfiability of SOL_1 explains the undecidability of SOL_4 reported in [4] and the undecidability of model checking of SOL_3 is linked to the undecidability of model checking for SOL_4. The results on propositional modular logics are, to the best of our knowledge, original. The decidability of PML_3 shows that the communication in combination with the modular operators is not necessarily undecidable. The decidability of SOL_2 is useful for applications in which interleaving semantics is relevant. Notice that, in light of Table 1, the undecidability of satisfiability seems generated either by the combination of second order quantifiers with $\top|\phi$, or by the combination of $\diamond$ and $\triangleright$. Undecidability of model checking seems generated by the presence of $\top \triangleright \phi$ in the context of undecidable satisfiability.

For future work we intend to extend these results for more complex frameworks, especially for the case of stochastic and probabilistic systems. We are interested in studying the decision problems for the case of Markovian Logics [8, 9]. These are complex logics for stochastic/probabilistic systems with modular properties. They can be used to specify properties of, e.g., stochastic-CCS processes [10] and they enjoy most of the properties established for modular logics.

References

1. J. van Benthem, *Language in action. Categories, Lambdas and Dynamic Logic.* Elsevier Science Publisher, 1991
2. J.A. Bergstra, A. Ponse, S.A. Smolka (eds.), *Handbook of Process Algebra.* Elsevier, 2001.
3. L. Caires and L. Cardelli, *A Spatial Logic for Concurrency (Part I).* Inf. and Comp. 186/2, 2003.
4. L. Caires and E. Lozes, *Elimination of Quantifiers and Decidability in Spatial Logics for Concurrency.* In Proc. of CONCUR'2004, LNCS 3170, 2004.
5. C. Calcagno, L. Cardelli and A. D. Gordon, *Deciding validity in a spatial logic for trees.* Journal of Functional Programming, 15, 2005.
6. C.Calcagno, et al. *Computability and complexity results for a spatial assertion language for data structures.* In Proc. of FSTTCS01, LNCS 2245, 2001.
7. L. Cardelli and A. D. Gordon. *Anytime, Anywhere: Modal Logics for Mobile Ambients.* In Proc. of 27th ACM Symposium on Principles of Programming Languages, 2000.
8. L. Cardelli, K. G. Larsen, R. Mardare. *Continuous Markovian Logics - From Complete Axiomatization to the Metric Space of Formulas.* In Proc. of Computer Science Logic CSL 2011.
9. L. Cardelli, K. G. Larsen, R. Mardare. *Modular Markovian Logic.* in Proc. ICALP 2011.
10. L. Cardelli, R. Mardare. *The Measurable Space of Stochastic Processes.* QEST 2010, IEEE Press, 2010.
11. W. Charatonik, J.M. Talbot, *The decidability of model checking mobile ambients.* In Proc. of CSL, LNCS 2142, 2001.
12. M. Dam, *Model checking mobile processes.* Inf. and Comp. 129(1), 1996.
13. V. Gyuris, *Associativity does not imply undecidability without the axiom of Modal Distribution.* In M. Marx, et.al eds., Arrow Logic and Multi-Modal Logic, CSLI and FOLLI, 1996.
14. M. Hennessy, R. Milner, *Algebraic laws for Nondeterminism and Concurrency.* JACM 32(1), 1985.

15. R. Mardare, *Logical analysis of Complex Systems. Dynamic Epistemic Spatial Logics.* PhD thesis, DIT, University of Trento, 2006.
16. R. Mardare, *Observing distributed computation. A dynamic-Epistemic approach.* In Proc. CALCO'07, LNCS 4624, 2007.
17. R. Mardare, C. Priami, *A logical approach to security in the context of Ambient Calculus.* Electronic Notes in Theoretical Computer Science, N 99:3-29, 2004.
18. R. Mardare, C. Priami. *A Propositional Branching Temporal Logic for the Ambient Calculus.* Tech.Rep. DIT-03-053, University of Trento, Italy, 2003.
19. R. Mardare, C. Priami, *Decidable extensions of Hennessy-Milner Logic.* In Proc. FORTE'06, LNCS 4229, 2006.
20. R. Mardare, A. Polocriti, *A Complete Axiomatic System for a Process-based Spatial Logic.* In Proc. MFCS'08, LNCS 5168, 2008.
21. R. Milner, *A Calculus of Communicating Systems.* Springer-Verlag New York, Inc., 1982.
22. R. Milner, J. Parrow and D. Walker, *Modal logics for mobile processes.* TCS 114, 1993.
23. D. Sangiorgi, *Extensionality and Intensionality of the Ambient Logics.* In Proc. of the 28th ACM Annual Symposium on Principles of Programming Languages, 2001.
24. C. Stirling, *Modal and temporal properties of processes.* Springer-Verlag New York, Inc., 2001.
25. A. Urquhart, *Semantics for Relevant Logics.* Journal of Symbolic Logic, 37(1), 1972.

Appendix

In this appendix we present some of the proofs of the main lemmas presented in the paper.

*Proof (**Proof of Lemma 2**).*

2. We prove it by induction on h. The case $h = 0$ is immediate.
For the case $h + 1$, suppose that $P \approx^{\Omega}_{(h+1,w)} P'$ and $Q \approx^{\Omega}_{(h+1,w)} Q'$.

Consider any $i = 1..w$, and any $\alpha \in \Omega$ such that $P|Q \equiv \alpha.R_1|...|\alpha.R_i|R_{i+1}$. Suppose, without loss of generality, that R_j are ordered in such a way that there exist $k \in 1..i$, P'', Q'' such that $P \equiv \alpha.R_1|...|\alpha.R_k|P''$, $Q \equiv \alpha.R_{k+1}|...|\alpha.R_i|Q''$ and $R_{i+1} \equiv P''|Q''$. Because $k \in 1..w$, from $P \approx^{\Omega}_{(h+1,w)} P'$ we have $P' \equiv \alpha.P'_1|...|\alpha.P'_k|P_0$ such that $R_j \approx^{\Omega}_{(h,w)} P'_j$ for $j = 1..k$. Similarly, from $Q \approx^{\Omega}_{(h+1,w)} Q'$ we have $Q' \equiv \alpha.Q'_{k+1}|...|\alpha.Q'_i|Q_0$ such that $R_j \approx^{\Omega}_{(h,w)} Q'_j$ for $j = (k+1)..i$. Hence, $P'|Q' \equiv \alpha.P'_1|...|\alpha.P'_k|\alpha.Q'_{k+1}|...|\alpha.Q'_i|P_0|Q_0$ with $R_j \approx^{\Omega}_{(h,w)} P'_j$ for $j = 1..k$ and $R_j \approx^{\Omega}_{(h,w)} Q'_j$ for $j = (k+1)..i$.

*Proof (**Proof of Lemma 4**).*

We prove it by induction on h. The case $h = 0$ is trivial.
The case $h + 1$: Suppose that $P'|P'' \approx^{\Omega}_{(h+1,w)} Q$. Let $w = w_1 + w_2$.

Following an idea proposed in [5], we say that a process P is in $\Omega_{(h,w)}$-*normal form* if whenever $P \equiv \alpha_1.P_1|\alpha_2.P_2|P_3$ for $\alpha_1, \alpha_2 \in \Omega$ and $P_1 \approx^{\Omega}_{(h,w)} P_2$ then $P_1 \equiv P_2$. Note that $P \approx^{\Omega}_{(h+1,w)} \alpha_1.P_1|\alpha_2.P_1|P_3$. This shows that for any P, any Ω and any (h, w) we can find a P_0 such that P_0 is in (h, w)-normal form and $P \approx^{\Omega}_{(h+1,w)} P_0$.

We can suppose, without loosing generality, that the canonical representations of P', P'' and Q are[3]: $P' \equiv (\alpha_1.P_1)^{k'_1}|...|(\alpha_n.P_n)^{k'_n}|P_1$, $P'' \equiv (\alpha_1.P_1)^{k''_1}|...|(\alpha_n.P_n)^{k''_n}|P_2$ and

[3] Else we can replace P', P'' with $(h + 1, w)$-related processes having the same (h, w)-normal forms

$Q \equiv (\alpha_1.P_1)^{l_1}|...|(\alpha_n.P_n)^{l_n}|Q_1$, where P_1, P_2, Q_1 have all the guarded subprocesses prefixed by actions that are not in Ω. For each $i \in 1..n$, we split $l_i = l'_i + l''_i$ in order to obtain a splitting of Q. We define the splitting of l_i such that $(\alpha_i.P_i)^{k'_i} \approx^{\Omega}_{h+1,w_1} (\alpha_i.P_i)^{l'_i}$ and $(\alpha_i.P_i)^{k''_i} \approx^{\Omega}_{h+1,w_2} (\alpha_i.P_i)^{l''_i}$. We do this as follows:

If $k'_i + k''_i < w_1 + w_2$ then $P'|P'' \approx^{\Omega}_{h+1,w} Q$ implies $l_i = k'_i + k''_i$, so we can choose $l'_i = k'_i$ and $l''_i = k''_i$.

If $k'_i + k''_i \geq w_1 + w_2$ then $P'|P'' \approx^{\Omega}_{h+1,w} Q$ implies $l_i \geq w_1 + w_2$. We meet the following subcases:

- $k'_i \geq w_1$ and $k''_i \geq w_2$. We choose $l'_i = w_1$ and $l''_i = l_i - w_1$ (note that as $l_i \geq w_1 + w_2$, we have $l''_i \geq w_2$).
- $k'_i < w_1$, then we must have $k''_i \geq w_2$. We choose $l'_i = k'_i$ and $l''_i = l_i - k'_i$. So $l''_i \geq w_2$ as $l_i \geq w_1 + w_2$ and $l'_i < w_1$.
- $k''_i < w_2$ is similar with the previous one. We choose $l''_i = k''_i$ and $l'_i = l_i - k''_i$.

Now, for $Q' \equiv (\alpha_1.P_1)^{l'_1}|...|(\alpha_n.P_n)^{l'_n}$ and $Q'' \equiv (\alpha_1.P_1)^{l''_1}|...|(\alpha_n.P_n)^{l''_n}$, the result is verified.

*Proof (**Proof of Lemma 5**).*

Because $P \approx^{\Omega}_{(h,w)} Q$, $\alpha \in \Omega$ and $P \equiv \alpha.P'|P''$, we obtain that $Q \equiv \alpha.Q'|Q''$ with $P' \approx^{\Omega}_{(h-1,w)} Q'$. We prove that $P'|P'' \approx^{\Omega}_{(h-1,w-1)} Q'|Q''$.

Consider $\beta \in \Omega$ and $i = 1..w-1$ such that: $P'|P'' \equiv \beta.P_1|...|\beta.P_i|P^\star$. We can suppose that, for some $k \leq i$, we have $P' \equiv \beta.P_1|...|\beta.P_k|P^+$, $P'' \equiv \beta.P_{k+1}|...|\beta.P_i|P^-$ and $P^\star \equiv P^+|P^-$. Because $P' \approx^{\Omega}_{(h-1,w)} Q'$ and $k \leq i \leq w-1$, we obtain that $Q' \equiv \beta.Q_1|...|\beta.Q_k|Q^+$ with $P_j \approx^{\Omega}_{(h-2,w)} Q_j$ for $j = 1..k$. Further we distinguish two cases.

1. If $\alpha \neq \beta$, then we have $P \equiv \beta.P_{k+1}|...|\beta.P_i|(P^-|\alpha.P')$ and because $P \approx^{\Omega}_{(h,w)} Q$, we obtain $Q \equiv \beta.R_{k+1}|...|\beta.R_i|R^\star$ with $R_j \approx^{\Omega}_{(h-1,w)} P_j$ for $j = k+1..i$. But $Q \equiv \alpha.Q'|Q''$ and because $\alpha \neq \beta$, we obtain $Q'' \equiv \beta.R_{k+1}|...|\beta.R_i|R^+$ that gives us in the end $Q'|Q'' \equiv \beta.Q_1|...|\beta.Q_k|\beta.R_{k+1}|...|\beta.R_i|(R^+|Q^+)$, with $P_j \approx^{\Omega}_{(h-2,w)} Q_j$ for $j = 1..k$ (hence, $P_j \approx^{\Omega}_{(h-2,w-1)} Q_j$) and $P_j \approx^{\Omega}_{(h-1,w)} R_j$ for $j = k+1..i$ (hence, $P_j \approx^{\Omega}_{(h-2,w-1)} R_j$).

2. If $\alpha = \beta$, then we have $P \equiv \alpha.P_{k+1}|...|\alpha.P_i|\alpha.P'|P^-$ and as $P \approx^{\Omega}_{(h,w)} Q$ and $i \leq w-1$, we obtain $Q \equiv \alpha.R_{k+1}|...|\alpha.R_i|\alpha.R'|R^\star$, with $R_j \approx^{\Omega}_{(h-1,w)} P_j$ for $j = k+1..i$ and $R' \approx^{\Omega}_{(h-1,w)} P'$. Because $P' \approx^{\Omega}_{(h-1,w)} Q'$ and $\approx^{\Omega}_{(h,w)}$ is an equivalence relation, we can suppose that[4] $R' \equiv Q'$. Consequently, $Q \equiv \alpha.R_{k+1}|...|\alpha.R_i|\alpha.Q'|R^\star$ that gives $Q'' \equiv \alpha.R_{k+1}|...|\alpha.R_i|R^\star$, which entails further $Q'|Q'' \equiv \alpha.Q_1|...|\alpha.Q_k|\alpha.R_{k+1}|...|\alpha.R_i|(R^\star|Q^+)$ with $P_j \approx^{\Omega}_{(h-2,w)} Q_j$ for $j = 1..k$ (hence, $P_j \approx^{\Omega}_{(h-2,w-1)} Q_j$) and $P_j \approx^{\Omega}_{(h-1,w)} R_j$ for $j = k+1..i$ (hence, $P_j \approx^{\Omega}_{(h-2,w-1)} R_j$).

All these prove that $P'|P'' \approx^{\Omega}_{(h-1,w-1)} Q'|Q''$.

The communication case goes similarly.

*Proof (**Proof of Lemma 6**).* We construct Q inductively on h. For the case $P \equiv 0$ we take $Q \equiv P$, as $0 \in \mathbb{P}^{\Omega^+}_{(h,w)}$.

[4] Indeed, if $\alpha.Q'$ is a subprocess of $R^\star$ then we can just substitute R' with Q'; if $\alpha.Q' \equiv \alpha.R_s$, then $Q' \approx^{\Omega}_{(h-1,w)} P_s$ and as $Q' \approx^{\Omega}_{(h-1,w)} P'$ and $P' \approx^{\Omega}_{(h-1,w)} R'$ we derive $R' \approx^{\Omega}_{(h-1,w)} P_s$ and $Q' \approx^{\Omega}_{(h-1,w)} P'$, so we can consider this correspondence.

Suppose $P \not\equiv 0$. Let $\beta = succ(sup(\Omega))$. In the case $h = 0$ we just take $Q \equiv \beta.0$.

The case $h + 1$. Suppose, without loss of generality, that

$$P \equiv (\alpha_1.P_1)^{k_1}|...|(\alpha_n.P_n)^{k_n}|(\gamma_{n+1}.P_{n+1})^{k_{n+1}}|...|(\gamma_{n+m}.P_{n+m})^{k_{n+m}}$$

where $\alpha_1, ..\alpha_n \in \Omega$ with $\alpha_i.P_i \not\equiv \alpha_j.P_j$ for $i \neq j$, and $\gamma_{n+1}, ..\gamma_{n+m} \in \Sigma \setminus \Omega$ with $\gamma_i.P_i \not\equiv \gamma_j.P_j$ for $i \neq j$.

Let P'_j for $j = 1..n$ be the processes constructed at the previous inductive step such that $P_j \approx^{\Omega}_{(h,w)} P'_j$ with $P'_j \in \mathbb{P}^{\Omega^+}_{(h,w)}$ - their existence is guaranteed by the inductive hypothesis. Let $l_i = min(k_i, w)$ and consider the process $P' \equiv (\alpha_1.P'_1)^{l_1}|...|(\alpha_n.P'_n)^{l_n}|\beta.0$. It is trivial to verify that P' is a process that fulfills the requirements of the lemma, i.e. $P \approx^{\Omega}_{(h,w)} P'$ and $P' \in \mathbb{P}^{\Omega^+}_{(h,w)}$.

*Proof (**Proof of Lemma 10**).* Induction on the structure of ϕ. We show here only the nontrivial cases.

The case $\phi = \langle\alpha\rangle\psi$: $P \models \langle\alpha\rangle\psi$ iff $P \xrightarrow{\alpha} P'$ and $P' \models \psi$. Suppose that $(\!|\psi|\!) = (h, w)$. Then $(\!|\phi|\!) = (h + 1, w + 1)$. Because $\alpha \in act(\phi)$ and $P \approx^{act(\phi)}_{(h+1,w+1)} Q$, we obtain applying Lemma 5 that $Q \xrightarrow{\alpha} Q'$ and $P' \approx^{act(\phi)}_{(h,w)} Q'$. We can apply the inductive hypothesis, as $P' \models \psi$ and we obtain $Q' \models \psi$. Then $Q \models \phi$.

The case $\phi = \langle\alpha, \overline{\alpha}\rangle\psi$: can be prove as the previous one using the second part of Lemma 5.

The case $\phi = \psi_1|\psi_2$: $P \models \psi_1|\psi_2$ iff $P \equiv S|R$, $S \models \phi_1$ and $R \models \psi_2$. Suppose that $(\!|\psi_1|\!) = (h_1, w_1)$ and $(\!|\psi_2|\!) = (h_2, w_2)$. Then $(\!|\phi|\!) = (max(h_1, h_2), w_1 + w_2)$. Applying Lemma 4 for $P \approx^{act(\phi)}_{(max(h_1,h_2),w_1+w_2)} Q$, we obtain that $Q \equiv S'|R'$ such that $S \approx^{act(\phi)}_{(max(h_1,h_2),w_1)} S'$ and $R \approx^{act(\phi)}_{(max(h_1,h_2),w_2)} R'$. Further Lemma 3 gives $S \approx^{act(\psi_1)}_{(h_1,w_1)} S'$ and $R \approx^{act(\psi_2)}_{(h_2,w_2)} R'$. Further, the inductive hypothesis gives $S' \models \psi_1$ and $R' \models \psi_2$, i.e. $Q \models \psi_1|\psi_2$.

The case $\phi = \psi_1 \triangleright \psi_2$: $P \models \psi_1 \triangleright \psi_2$ iff any $R \models \psi_1$ implies $P|R \models \psi_2$. But $P \approx^{act(\phi)}_{(\!|\phi|\!)} Q$ and $R \approx^{act(\phi)}_{(\!|\phi|\!)} R$ implies, by Lemma 2, that $P|R \approx^{act(\phi)}_{(\!|\phi|\!)} Q|R$. Further $P|R \approx^{act(\psi_2)}_{(\!|\psi_2|\!)} Q|R$ and because $P|R \models \psi_2$, we can apply the inductive hypothesis deriving $Q|R \models \psi_2$. Hence, $R \models \psi_1$ implies $Q|R \models \psi_2$, i.e., $Q \models \psi_1 \triangleright \psi_2$.