Available online at www.sciencedirect.com

SciVerse ScienceDirect

Electronic Notes in Theoretical Computer Science

Electronic Notes in Theoretical Computer Science 278 (2011) 229–243

www.elsevier.com/locate/entcs

An Epistemic Predicate CTL^* for Finite Control π-Processes

Dimitar P. Guelev[1]

Institute of Mathematics and Informatics
Bulgarian Academy of Sciences
Sofia, Bulgaria

Mads Dam[2]

School of Computer Science and Communication
Royal Institute of Technology
Stockholm, Sweden

Abstract

We examine model checking of finite control π-calculus processes against specifications in epistemic predicate CTL^*. In contrast to branching time settings such as CTL or the modal μ-calculus, the general problem, even for LTL, is undecidable, essentially because a process can use the environment as unbounded storage. To circumvent this problem attention is restricted to closed processes for which internal communication along a given set of known channels is observable. This allows to model processes operating in a suitably memory-bounded environment. We propose an epistemic predicate full CTL^* with perfect recall which is interpreted on the computation trees defined by such finite control π-calculus processes. We demonstrate the decidability of model-checking by a reduction to the decidability of validity in quantified full propositional CTL^*.

Keywords: epistemic temporal logic, pi-calculus, model checking

Introduction

The π-calculus [12,16] has attracted a lot of interest as a computational model for distributed systems. Along with most other process algebras the calculus is Turing-complete in general. Therefore most interesting decision problems about the π-calculus are undecidable. Algorithmic support mainly applies to its *finite-control* subset, where the use of parallel composition is syntactically restricted.

Epistemic extensions of temporal logic have proved highly valuable to express properties of agents' evolving knowledge in distributed systems [6]. The π-calculus

[1] Email: gelevdp@math.bas.bg

[2] Email: mfd@kth.se

1571-0661
doi:10.1016/j.entcs.2011.10.018

extends the established computational models of epistemic TLs by the possibility to dynamically create new communication channels. It is of interest to examine how this feature can be accomodated within the epistemic logic framework.

In this paper we introduce a system of predicate epistemic CTL^* on the computation trees of finite control π-calculus processes. Our epistemic operator conforms with the established view that a fact is known if it is true about all the computations which the knower finds identical to the actual one. Epistemic TLs refer to agent identity and "knowers". The π-calculus does not have these notions, but epistemic modalities can be interpreted on π-processes in other ways. Cohen and Dam [4] interpret the epistemic modality in terms of static equivalence [1], but their work addresses only static knowledge. Chadha et al [3] suggest a single knower epistemic TL for π-processes based on a form of trace equivalence. However, it is unclear how this extends to multiple agents, and why $\Box$ and $\boxminus$ are the only *temporal* operators considered. In the experiment reported here we take a different approach: We identify knowers with their observational power, which is determined by a set of initially "known", or tapped, channels. This set grows by adding the channel names which become communicated along the channels already tapped. We write $\mathsf{K}_{x_1,\ldots,x_n}\varphi$ for *a knower who initially taps* $x_1,\ldots,x_n$ *knows that* φ.

Directly extending the known decidability results for the pure branching time case [5] to just linear time temporal logic LTL, let alone an epistemic extension of CTL^* is, however, not possible. Even with the restriction to finite control, exchange with an external environment renders model-checking of LTL properties unsolvable, because of the possibility to restrict the environment to behave as storage for a given Turing machine's tape, and to state that the machine never terminates. A proof is sketched at the end of the paper. The undecidability carries over to (epistemic extensions of) CTL^*. To side-step this complication, we constrain the environment by shifting attention to closed systems, and assume instead that knowers observe only internal communication along the set of tapped names, communication along which is observable. The upshot is that processes can be predicated only when placed in a fixed finite closing environment.

We prove the decidability of model-checking for our system. We encode the execution tree of the given π-process as a finite Kripke frame and reduce the model-checking of any given predicate epistemic CTL^* formula φ on this tree to the satisfiability of a translation of φ into quantified propositional CTL^* ($QCTL^*$) on trees, which is known to be decidable from [8,9].

1 Background on π-Calculus

Finite control π-terms syntax can be given by the BNF

$$P ::= 0 \mid \alpha.P \mid (\nu y)P \mid P + P \mid \textbf{if } x = y \textbf{ then } P \textbf{ else } P \mid p(y,\ldots,y)$$
$$Q ::= 0 \mid P \mid Q|Q \mid (\nu x)Q$$

Here P, Q are process terms that use (channel) names x, y for communication. A communication action α is either the input of a name y along a channel named x, written as $x(y)$, or the output of y along x, written $\overline{x}y$, or the neutral, unobservable action τ. Names can be locally scoped by the operator (νy) which prevents communication along y (but allows y to be passed as a parameter, resulting in so-called scope extrusion of y, as detailed below). Other operators are action prefixing, choice $(+)$, conditionals, and parallel composition. A process is a term of the form Q together with a finite set of definitions of the form $p(x_1, \ldots, x_n) = P$, for the recursive invocations in Q and in the definitions' own righthand sides. Below we elide the distinction between single process terms P and parallel compositions Q, and use P to range over both. The set of all names in a π-term P is denoted by $n(P)$. The sets of free and bound names are written $fn(P)$ and $bn(P)$, respectively, the binders being (νx) and the input prefix $x(y)$, which binds x, resp. y. Binders induce a relation of structural congruence $\equiv$ on terms, including α-conversion, briefly detailed below.

We consider only executions

$$P^0 \xrightarrow{\tau}_{C^1} P^1 \xrightarrow{\tau}_{C^2} \cdots \xrightarrow{\tau}_{C^k} P^k \xrightarrow{\tau}_{C^{k+1}} \cdots \tag{1}$$

which consist entirely of silent steps, in order to prevent environment interactions, as explained in the introduction. Transitions are annotated by the sets C^k of internal communication acts which are possibly observed by knowers. Each P^k has the form

$$(\nu x_1) \ldots (\nu x_m) P \tag{2}$$

where P has no occurrences of ν. This form can be achieved using structural congruence. Annotations C^k consist of communication acts written in the form $c(x)$. Annotated transitions are derived by the following axioms and rules, a variant of the so-called *early semantics* of the π-calculus, cf. [14]:

$$\tau.P \xrightarrow{\tau}_{\emptyset} P \qquad x(y).P \xrightarrow{x(z)}_{\emptyset} [z/y]P \qquad \overline{x}y.P \xrightarrow{\overline{x}y}_{\emptyset} P$$

$$\frac{P \xrightarrow{\alpha}_C P' \quad y \notin n(\alpha) \quad y \notin n(C)}{(\nu y)P \xrightarrow{\alpha}_C (\nu y)P'} \qquad \frac{P \xrightarrow{\alpha}_C P' \quad y \in n(C)}{(\nu y)P \xrightarrow{\alpha}_C P'} \qquad \frac{P \xrightarrow{\alpha}_C P'}{P + Q \xrightarrow{\alpha}_C P'}$$

$$\frac{P_1 \xrightarrow{\alpha}_C P_1'}{\textbf{if } x = x \textbf{ then } P_1 \textbf{ else } P_2 \xrightarrow{\alpha}_C P_1'} \qquad \frac{P_2 \xrightarrow{\alpha}_C P_2' \qquad x \neq y}{\textbf{if } x = y \textbf{ then } P_1 \textbf{ else } P_2 \xrightarrow{\alpha}_C P_2'}$$

$$\frac{Q_1 \xrightarrow{\alpha}_C Q_1' \qquad bn(\alpha) \cap fn(Q_1) = \emptyset}{Q_1|Q_2 \xrightarrow{\alpha}_C Q_1'|Q_2} \qquad \frac{Q_1 \xrightarrow{\overline{x}y}_{\emptyset} Q_1' \qquad Q_2 \xrightarrow{x(y)}_{\emptyset} Q_2'}{Q_1|Q_2 \xrightarrow{\tau}_{\{x(y)\}} Q_1'|Q_2'}$$

$$\text{(Congruence)} \qquad \frac{P \xrightarrow{\alpha}_C Q \qquad P \equiv P' \qquad Q \equiv Q'}{P' \xrightarrow{\alpha}_C Q'}$$

Symmetric rules for $+$ and parallel composition $|$ are derivable using structural congruence. Annotations can be either $\emptyset$, or singletons. Together with the identities $A|(\nu x)B \equiv (\nu x)(A|B)$, $x \notin fv(A)$, and $p(x_1, \ldots, x_n) \equiv P$, given $p(x_1, \ldots, x_n) = P$, the congruence rule allows to avoid the use of bound output action $\overline{x}(y)$, and a

dedicated rule about recursive invocations. It is possible to show that $P \xrightarrow{\tau}_{\{x(y)\}} Q$ according to the above semantics iff $P \xrightarrow{\tau} (\nu x)(\nu y)Q$ according to the early semantics of [14], where one or both of (νx) or (νy) may be absent.

2 Epistemic Predicate Full CTL^* on Finite Control π-Processes

Using α-conversion it is easy to write executions such as (1) in such a manner that names are never reused in the following sense.

Definition 2.1 The *extent (lifetime)* of name x in an execution E written as in (1) is the set $L_E(x) \stackrel{def}{=} \{k < \omega : x \in n(P^k) \cup n(C^k)\}$. E is *standard* if, for every x, $L_E(x)$ is either $\emptyset$, or a finite or infinite interval.

A model for $EPCTL^*$ is the Kripke frame $\mathcal{T}(P^0)$ whose paths correspond to the standard executions starting from some given π-term P^0. Fix a countably infinite set D including all names in $\mathcal{T}(P^0)$.

Definition 2.2 $\mathcal{T}(P^0) = \langle W, R\rangle$ where W consists of all the pairs of the form $\langle P, C\rangle$ where P is a process term of the form (2) that occurs in some execution starting from P^0, and $C \in \{\emptyset\} \cup \{c(c') : c, c' \in D\}$. $\langle P', C'\rangle R \langle P'', C''\rangle$ iff either $P' \xrightarrow{\tau}_{C''} P''$, or $P' = P''$, $C'' = \emptyset$ and P' is either deadlocked or terminated.

The condition $\langle P, C\rangle R\langle P, \emptyset\rangle$ for terminated and deadlocked P rules out finite maximal paths in $\mathcal{T}$.

Given P^0, there exists a finite set $\mathbf{P}$ of ν-free process terms such that the following condition holds: Let $\{y_1, \ldots, y_N\} = \bigcup_{P\in\mathbf{P}} n(P)$ and let $\mathbf{A}$ be the set $\{\emptyset\} \cup \{\{y_i(y_j)\} : i, j = 1, \ldots, N\}$ of annotations written using $y_1, \ldots, y_N$. Then all the annotated silent transitions $P^k \xrightarrow{\tau}_{C^{k+1}} P^{k+1}$ in executions (1) starting with P^0 can be written in the form

$$\sigma(\nu u_1)\ldots(\nu u_r)Q' \xrightarrow{\tau}_{\sigma B} \sigma(\nu v_1)\ldots(\nu v_s)Q'' \tag{3}$$

where $Q', Q'' \in \mathbf{P}$, $u_1, \ldots, u_r, v_1, \ldots, v_s \in \{y_1, \ldots, y_N\}$, $B \in \mathbf{A}$, $\sigma \stackrel{def}{=} [\![n_1/y_1, \ldots, n_N/y_N]\!]$ is the substitution of $y_1, \ldots, y_N$, by the pairwise distinct names $n_1, \ldots, n_N$, and $\sigma B \stackrel{def}{=} \{n_{j_1}(n_{j_2}) : y_{j_1}(y_{j_2}) \in B, j_1, j_2 = 1, \ldots, N\}$. We write σ using $[\![.]\!]$ and not $[.]$ to indicate that it affects the *bound* occurrences of $y_1, \ldots, y_N$ too. Since $n_1, \ldots, n_N$ are required to be distinct, our use of $[\![.]\!]$ is semantically correct. In particular, (3) is a derivable transition iff $(\nu u_1)\ldots(\nu u_r)Q' \xrightarrow{\tau}_B (\nu v_1)\ldots(\nu v_s)Q''$ is.

We use $\mathbf{P}$ as a vocabulary of predicate symbols for $\mathcal{T} = \mathcal{T}(P^0)$. Each $P \in \mathbf{P}$ is used as a $|fn(P)|$-ary predicate symbol. (Note that here P ranges over the ν-free parts of terms in the form (2). The only bound names of P can be the ys in the scope of an $x(y)$.) Given $\{z_1, \ldots, z_{|fn(P)|}\} \stackrel{def}{=} fn(P) \subseteq \{y_1, \ldots, y_N\}$, we fix the ordering $z_1, \ldots, z_{|fn(P)|}$, and, for any $n_1, \ldots, n_{|fn(P)|} \in D$, we define $P^{\mathcal{T}}(n_1, \ldots, n_{|fn(P)|})$ to

hold $\langle Q, A\rangle \in W$ iff $Q = [n_1/z_1, \ldots, n_{|fn(P)|}/z_{|fn(P)|}]P$. Similarly, we introduce a binary predicate symbol C for latest communication act, and a temporal proposition T for silent transitions.

This vocabulary may be inconvenient for immediate use, but with existential quantification and disjunction one can easily define predicates like, e.g., $Z(n_1, n_2)$ for *there exist a name y such that the current process term is of the form ... $|$ $n_1(x).p(n_2, x, y)$ $|$ ...*.

For an annotated execution E written as (1), the set $C_E(a, k)$ of the channels that are tapped by knower a at step k is defined as follows. $C_E(a, 0)$ is presumed to be predefined and the same for all E. Given $C_E(a, k)$, we put

$$C_E(a, k+1) \stackrel{def}{=} C_E(a, k) \cup \{c' : c(c') \in C^{k+1}, c \in C_E(a, k)\}. \tag{4}$$

In words, once a observes the communication of channel name c', communication over c' becomes observable to a too. Given $C_E(a, k)$, $k < \omega$, and two more executions $F_i = Q_i^0 \xrightarrow{\tau}_{A_i^1} \cdots \xrightarrow{\tau}_{A_i^k} Q_i^k \xrightarrow{\tau}_{A_i^{k+1}} \cdots$, $i = 1, 2$, we define $F_1 \sim_{a,k,E} F_2$ as the equivalence relation

$$(\forall j \le k)(\forall c \in C_E(a, j))(\forall c' \in D)(c(c') \in A_1^j \leftrightarrow c(c') \in A_2^j).$$

In words, $F_1 \sim_{a,k,E} F_2$ iff F_1 and F_2 have the same communication over channels that are observed by a in E at all steps $j \le k$. Since $F_1 \sim_{a,k,F_1} F_2$ entails $C_{F_2}(a, j) = C_{F_1}(a, j)$ for $j \le k$, and therefore $F_1 \sim_{a,k,F_1} F_2$ and $F_1 \sim_{a,k,F_2} F_2$ are equivalent, and $\sim_{a,k} \stackrel{def}{=} \lambda F_1 F_2.F_1 \sim_{a,k,F_1} F_2$ is an equivalence relation. F_1 and F_2 are indiscernible to a until step k iff $F_1 \sim_{a,k} F_2$. We define our epistemic modality by means of $\sim_{a,k}$.

The syntax of *EPCTL** is

$$\varphi ::= \bot \mid P(x, \ldots, x) \mid \varphi \Rightarrow \varphi \mid \exists x\varphi \mid \ominus\varphi \mid \bigcirc\varphi \mid (\varphi \mathsf{S} \varphi) \mid (\varphi \mathsf{U} \varphi) \mid \exists\varphi \mid \mathsf{K}_{x,\ldots,x}\varphi$$

where the occurrences of x represent individual variables. The counterparts of standard executions in $\mathcal{T}$ are standard R-paths.

Definition 2.3 An infinite sequence

$$\rho = \langle P^0, C^0\rangle, \ldots, \langle P^k, C^k\rangle, \ldots \in W^\omega \tag{5}$$

is a *standard R-path* if P^0 is the process term used to define $\mathcal{T} = \mathcal{T}(P^0)$, $C^0 = \emptyset$, $\langle P^k, C^k\rangle R \langle P^{k+1}, C^{k+1}\rangle$ for all $k < \omega$ and the corresponding execution (1) is standard. Given R-paths ρ_1 and ρ_2 and channels $c_1, \ldots, c_m \in D$, we write $\rho_1 \sim_{c_1,\ldots,c_m,k} \rho_2$ if $E_1 \sim_{a,k} E_2$ for the corresponding executions E_1 and E_2, and a such that $\{c_1, \ldots, c_m\} = C_{E_1}(a, 0) = C_{E_2}(a, 0)$.

Definition 2.4 Given a standard R-path (5), a valuation v of the individual vari-

ables into D, $k < \omega$ and a formula φ, $\mathcal{T}, v, \rho, k \models \varphi$ is defined by the clauses

$$
\begin{array}{ll}
\mathcal{T}, v, \rho, k \not\models \bot & \\
\mathcal{T}, v, \rho, k \models P(x_1, \ldots, x_{|fn(P)|}) & \text{iff } P^k \text{ is } [v(x_1)/z_1, \ldots, v(x_{|fn(P)|})/z_{|fn(P)|}]P \\
\mathcal{T}, v, \rho, k \models C(x_1, x_2) & \text{iff } C^k = \{v(x_1)(v(x_2))\} \\
\mathcal{T}, v, \rho, k \models T & \text{iff } C^k = \emptyset \\
\mathcal{T}, v, \rho, k \models \varphi \Rightarrow \psi & \text{iff either } \mathcal{T}, v, \rho, k \not\models \varphi, \text{ or } \mathcal{T}, v, \rho, k \models \psi \\
\mathcal{T}, v, \rho, k \models \exists x\varphi & \text{iff } \mathcal{T}, v[x \mapsto d], \rho, k \models \varphi \text{ for some } d \in D \\
\mathcal{T}, v, \rho, k \models \ominus\varphi & \text{iff } k > 0 \text{ and } \mathcal{T}, v, \rho, k-1 \models \varphi \\
\mathcal{T}, v, \rho, k \models \bigcirc\varphi & \text{iff } \mathcal{T}, v, \rho, k+1 \models \varphi \\
\mathcal{T}, v, \rho, k \models (\varphi \mathsf{S} \psi) & \text{iff there exists an } n \leq k \text{ s.t. } \mathcal{T}, v, \rho, k-n \models \psi \\
 & \quad \text{and } \mathcal{T}, v, \rho, k-j \models \varphi \text{ for } j = 0, \ldots, n-1 \\
\mathcal{T}, v, \rho, k \models (\varphi \mathsf{U} \psi) & \text{iff there exists an } n < \omega \text{ s.t. } \mathcal{T}, v, \rho, k+n \models \psi \\
 & \quad \text{and } \mathcal{T}, v, \rho, k+j \models \varphi \text{ for } j = 0, \ldots, n-1
\end{array}
$$

$$
\begin{array}{ll}
\mathcal{T}, v, \rho, k \models \exists\varphi & \text{iff there exists a standard } R\text{-path } \rho' \\
 & \quad \text{s.t. } \rho'[0..k] = \rho[0..k] \text{ and } \mathcal{T}, v, \rho', k \models \varphi \\
\mathcal{T}, v, \rho, k \models \mathsf{K}_{x_1,\ldots,x_m}\varphi & \text{iff } \mathcal{T}, v, \rho', k \models \varphi \text{ for all standard} \\
 & \quad R\text{-paths } \rho' \text{ s.t. } \rho \sim_{v(x_1),\ldots,v(x_m),k} \rho'
\end{array}
$$

Here $\rho[0..k]$ stands for the finite prefix of ρ of length $k+1$. As expected, $FV(\mathsf{K}_{x_1,\ldots,x_m}\varphi) = FV(\varphi) \cup \{x_1, \ldots, x_m\}$.

We use $\top$, $\neg$, $\wedge$, $\vee$ and $\Leftrightarrow$ as abbreviations in the usual way; I, $\diamondminus\varphi$, $\boxminus\varphi$, $\Diamond\varphi$, $\Box\varphi$, $(\varphi \mathsf{W} \psi)$ and $(\varphi \mathsf{V} \psi)$ abbreviate the formulas $\neg\ominus\top$, $(\top \mathsf{S} \varphi)$, $\neg\diamondminus\neg\varphi$, $(\top \mathsf{U} \varphi)$, $\neg\Diamond\neg\varphi$, $(\varphi \mathsf{U} \psi) \vee \Box\varphi$ and $(\varphi \mathsf{S} \psi) \vee \boxminus\varphi$, respectively.

Example 2.5 Let $P^0 = p(c)|q(c)$ where

$$p(x) = \overline{x}x.p(x) + (\nu y)\overline{x}y.p(y),\ q(x) = x(y).\textbf{if } x = y \textbf{ then } \mathbf{0} \textbf{ else } q(y).$$

A knower who can initially tap c is in a position to detect the termination of the right operand of $|$ in the process as soon as a tapped channel's name becomes transmitted along that same channel:

$$\mathcal{T}(P^0), P^0, v, 0 \models \forall x \forall w (C(x, w) \Rightarrow \forall\Box(\exists z \underline{p(z)|\mathbf{0}} \Rightarrow \mathsf{K}_x(\exists z \underline{p(z)|\mathbf{0}})),$$

where the atomic formula $p(z)|\mathbf{0}$ is underlined for better readability. To achieve this, the knower must follow the communication along the new channels y introduced

at each step. (Each of these channels is used once to announce the name of its successor, and then "forgotten" by the process.)

3 From $EPCTL^*$ on finite control π-processes to $QCTL^*$ on trees

Consider standard annotated executions (1) with process terms of the form (2) and the representation (3) of transitions in such executions again. The representation (3) applies if we allow some of $n_1, \ldots, n_N$ to be the auxiliary symbol $* \notin D$ too, provided that $n_j = *$ only if $y_j \notin n(Q') \cup n(Q'')$. To facilitate the presentation, in the sequel we use (3) with $n_1, \ldots, n_N$ ranging over $D \cup \{*\}$ and put $\sigma = [\ldots, */y_j, \ldots]$ instead of $y_j \notin \mathrm{dom}\sigma$.

We fix P^0, $\mathbf{P}$, D, $\{y_1, \ldots, y_N\} = \bigcup_{Q \in \mathbf{P}} n(Q)$ and $\mathbf{A}$ for the rest of the section. Given these, an annotated execution E of the form (1) can be written as

$$\sigma_0 Q^0 \xrightarrow{\tau}_{\sigma_1 B^1} \cdots \xrightarrow{\tau}_{\sigma_k B^k} \sigma_k Q^k \xrightarrow{\tau}_{\sigma_{k+1} B^{k+1}} \cdots \tag{6}$$

where $Q^k \in \mathbf{P}$, $B^{k+1} \in \mathbf{A}$ and σ_k are substitutions as above which satisfy $\sigma_{k+1} Q^k = \sigma_k Q^k$, and the additional condition $\mathrm{ran}\sigma_k \setminus \{*\} = n(\sigma_k Q^k) \cup n(\sigma_k B^k) = n(P^k) \cup n(C^k)$ for all k. Then obviously $L_E(n) = \{k < \omega : n \in \mathrm{ran}\sigma_k\}$. In the sequel we additionally require that if $\sigma_{k_1}(y_{i_1}) = \sigma_{k_2}(y_{i_2}) \neq *$ in the form (6) of E, then $i_1 = i_2$, for all $k_1, k_2 \in L_E(\sigma_{k_1}(y_{i_1}))$, that is, a name n should occupy the same slot y throughout its lifetime in E.

Up to a permutation of D, (6) is determined by the sequences Q^k, $k < \omega$, and B^k, $1 \leq k < \omega$, and, for each $j = 1, \ldots, N$, the steps k at which

$$\sigma_{k-1}(y_j) \neq \sigma_k(y_j). \tag{7}$$

To realise that, observe that in standard executions (7) is equivalent to $k = \min L_E(\sigma_k(y_j))$ and to $k - 1 = \max L_E(\sigma_{k-1}(y_j))$, provided that $\sigma_k(y_j) \neq *$ and $\sigma_{k-1}(y_j) \neq *$, respectively. Consequently, up to permutations of names, the standard executions starting from a given P^0 can be described by means of the finite Kripke frame $F = \langle W, R, w_0 \rangle$ with state space $W \stackrel{def}{=} \mathbf{P} \times \mathbf{A} \times \mathcal{P}(\{y_1, \ldots, y_N\})$, initial state $w_0 \stackrel{def}{=} \langle P^0, \emptyset, n(P^0) \rangle$ and transition relation R such that $\langle P', B', Y' \rangle R \langle P'', B'', Y'' \rangle$ iff $Y'' = (n(P'') \cup n(B'')) \triangle (n(P') \cup n(B'))$ and either $P' \xrightarrow{\tau}_{B''} P''$ is a derivable transition, or $P' = P'', B'' = \emptyset$ and P' is either deadlocked or terminated. Here $n(\emptyset) \stackrel{def}{=} \emptyset$, $n(\{y_{j_1}(y_{j_2})\}) \stackrel{def}{=} \{y_{j_1}, y_{j_2}\}$ and $A \triangle B \stackrel{def}{=} A \setminus B \cup B \setminus A$, as expected. The component Y of $\langle P, B, Y \rangle \in W$ is meant to denote the names from among $y_1, \ldots, y_N$, which disappear or (re)appear upon incoming transitions, respectively.

We use F to model-check the tree of all standard executions starting from P^0 for $EPCTL^*$ properties. Instead of immediately interpreting $EPCTL^*$ formulas on F, we use a propositional LTL formula $\mathcal{E}$ which describes the set of paths of F. To this

end introduce a finite vocabulary $\mathbf{L} = \{q_1, \ldots, q_K\}$ and a *valuation* $V : W \to \mathcal{P}(\mathbf{L})$. No connection between the values of the variables from $\mathbf{L}$ and the structure of the states of F is assumed. We only require V to satisfy $V(w') \neq V(w'')$ whenever $w' \neq w''$, which can be achieved iff $K \geq \log_2 |W|$. Given a state $w \in W$, let $\widehat{w} \stackrel{def}{=} \bigwedge_{q \in V(w)} q_i \wedge \bigwedge_{q \in \mathbf{L} \setminus V(w)} \neg q$. We put

$$\mathcal{E} \rightleftharpoons \widehat{w_0} \wedge \bigwedge_{w \in W} \Box(\widehat{w} \Rightarrow \bigcirc \bigvee_{w' \in R(w)} \widehat{w'}). \tag{8}$$

Now the validity of an arbitrary $QCTL^*$ formula φ in M is equivalent to $\models_{QCTL^*} \forall\mathcal{E} \Rightarrow \varphi$. By ch_j, busy_j, $\mathsf{comm}_{j,k}$, $j, k = 1, \ldots, N$, and tau, we denote boolean combinations of $q_1, \ldots, q_K$ which, up to equivalence, are determined by the following conditions, where $M = \langle W, R, w_0, V\rangle$ and $w = \langle Q, B, Y\rangle$:

$$M, w \models \mathsf{ch}_j \text{ iff } y_j \in Y \qquad M, w \models \mathsf{busy}_j \text{ iff } y_j \in n(Q) \cup n(B)$$
$$M, w \models \mathsf{tau} \text{ iff } B = \emptyset \qquad M, w \models \mathsf{comm}_{j_1,j_2} \text{ iff } B = \{y_{j_1}(y_{j_2})\}$$

The intended meaning of ch_j is to indicate that the occupation of y_j was changed upon the incoming transition, i.e., either $k = 0$, or $\sigma_{k-1}(y_j) \neq \sigma_k(y_j)$ in the representation (6) of executions; busy_j means that y_j currently holds a name and not $*$; tau means that the incoming transition was τ, and comm_{j_1,j_2} means that the incoming transition was $\sigma_k(y_{j_1})(\sigma_k(y_{j_2}))$.

Given $P \in \mathbf{P}$ and a sequence of indices $j_1, \ldots, j_{|fn(P)|} \in \{1, \ldots, N\}$, $P_{j_1,\ldots,j_{|fn(P)|}}$ denotes some boolean combination of $q_1, \ldots, q_K$ such that $M, \langle Q, B, Y\rangle \models P_{j_1,\ldots,j_{|fn(P)|}}$ iff Q is $[y_{j_1}/z_1, \ldots, y_{j_{|fn(P)|}}/z_{|fn(P)|}]P$ where $z_1, \ldots, z_{|fn(P)|}$ is the fixed ordering of $fn(P)$ previously associated with P.

Next we describe a translation $\mathsf{t}(.)$ of $EPCTL^*$ into $QCTL^*$ on tree Kripke models. Tree models allow the values of bound propositional variables to vary unrestrictedly along paths, whereas repeated occurrences of states along paths in non-tree models constrain the values of quantified variables at the respective positions to be the same too. By abuse of notation, we write $M = \langle W, R, w_0, V\rangle$ for the result of the unravelling of the finite Kripke model M described above into a tree one too. $QCTL^*$ extends propositional CTL^* by formulas of the form $\exists q\varphi$. $M, \rho, k \models \exists q\varphi$ holds iff there exists a $V' : W \to \mathcal{P}(\mathbf{L})$ such that $V'(p) = V(p)$ for $p \neq q$ and $\langle W, R, w_0, V'\rangle, \rho, k \models \varphi$.

The $QCTL^*$ translation $\mathsf{t}(\varphi)$ of an $EPCTL^*$ sentence φ satisfies $\models_{QCTL^*} \forall\mathcal{E} \Rightarrow \mathsf{t}(\varphi)$ where $\mathcal{E}$ is as in (8) iff φ is true about all the executions starting with a fixed P^0. As mentioned above, $\mathcal{E}$ allows the appearance of names in E to be determined up to a permutation on D. Since we assume φ to be a sentence, this is sufficient.

To handle quantification over names in $EPCTL^*$ we augment the description of the possible executions E which can be derived from $\mathcal{E}$ with a description of the identities between the names which appear in E and the values of the (bound) variables of φ. Without loss of generality we assume that no individual variable in φ is bound by more than one occurrence of $\exists$. Let $x_1, \ldots, x_M$ be all the individual

variables of φ. To describe the occurrences of $v(x_l)$ in an execution E for the relevant v, we take the form (6) of E and introduce the propositional variables $p_{j,l}$, $j = 1, \ldots, N$. The intended meaning of $p_{j,l}$ at step k is $v(x_l) = \sigma_k(y_j)$. As it becomes clear below, this enables translating $P(x_{l_1}, \ldots, x_{l_m})$ into $\bigvee\limits_{j_1,\ldots,j_m} P_{j_1,\ldots,j_m} \wedge p_{j_1,l_1} \wedge \ldots p_{j_m,l_m}$.

The translation of a formula of the form $\exists x_l \psi$ includes a formula of the form $\exists p_{1,l} \ldots \exists p_{N,l}(\mathcal{V}_l \wedge \mathsf{t}(\psi))$, in which $\mathcal{V}_l$ constrains $p_{j,l}$ to mark some possible extent $L_E(v(x_l)) = L_E(\sigma_k(y_j))$ of $v(x_l)$ in the executions E which correspond to the paths in $\mathcal{T}$ and in the corresponding $QCTL^*$ model M. The case of $p_{j,l}$ being satisfied nowhere along the given path corresponds to the name $v(x_l)$ appearing nowhere in E. Let

$$F_{j,l} \rightleftharpoons p_{j,l} \wedge \mathsf{busy}_j \wedge \bigwedge_{j' \neq j} \neg p_{j',l} \wedge \bigwedge_{l' \neq l} \neg p_{j,l'}.$$

$F_{j,l}$ means that x_l evaluates to $\sigma_k(y_j)$ at time k, and j is the only one with this property, and no other individual variable evaluates to $\sigma_k(y_j)$ at time k. The latter condition is included to simplify the handling of atomic formulas built using $=$. To express that x_l evaluates to none of the names $\sigma_k(y_j)$, we use the formula $G_l \rightleftharpoons \bigwedge\limits_{j=1}^{N} \neg p_{j,l}$. Using $F_{j,l}$ and G_l, we write

$$H_{j,l} \rightleftharpoons (G_l \mathsf{W} \mathsf{ch}_j \wedge F_{j,l} \wedge \bigcirc(F_{j,l} \wedge \neg \mathsf{ch}_j \mathsf{W} \mathsf{ch}_j \wedge \Box G_l)).$$

The satisfaction of $H_{j,l}$ at step 0 means that either $L_E(v(x_l)) = \emptyset$, or there exists a k such that $\sigma_k(y_j) \neq *$ for some k and $v(x_l) = \sigma_{k'}(y_j)$ for $k' \in L_E(v(x_l)) = L_E(\sigma_k(y_j))$. Now we can put $\mathcal{V}_l \rightleftharpoons \Diamond(\mathsf{I} \wedge \forall \bigvee\limits_{j=1}^{N} H_{j,l})$. The clauses for the translation, except that for epistemic formulas, are as follows:

$$\begin{aligned}
&\mathsf{t}(\bot) && \rightleftharpoons \bot \\
&\mathsf{t}(x_{l_1} = x_{l_2}) && \rightleftharpoons \bot \text{ if } l_1 \neq l_2 \\
&\mathsf{t}(x_l = x_l) && \rightleftharpoons \top \\
&\mathsf{t}(P(x_{l_1}, \ldots, x_{l_m})) && \rightleftharpoons \bigvee_{j_1,\ldots,j_m} \left(P_{j_1,\ldots,j_m} \wedge \bigwedge_{i=1}^{m} p_{j_i,l_i} \right) \\
&\mathsf{t}(C(x_{l_1}, x_{l_2})) && \rightleftharpoons \bigvee_{j_1,j_2} (\mathsf{comm}_{j_1,j_2} \wedge p_{j_1,l_1} \wedge p_{j_2,l_2}) \\
&\mathsf{t}(T) && \rightleftharpoons \mathsf{tau} \\
&\mathsf{t}(\mathsf{X}\varphi) && \rightleftharpoons \mathsf{X}\mathsf{t}(\varphi) \text{ for } \mathsf{X} \in \{\bigcirc, \ominus, \exists\} \\
&\mathsf{t}((\varphi \mathsf{X} \psi)) && \rightleftharpoons (\mathsf{t}(\varphi) \mathsf{X} \mathsf{t}(\psi)) \text{ for } \mathsf{X} \in \{\mathsf{U}, \mathsf{S}, \Rightarrow\} \\
&\mathsf{t}(\exists x_l \varphi) \rightleftharpoons \bigvee_{z \in FV(\exists x_l \varphi)} && \mathsf{t}([z/x_l]\varphi) \vee \exists p_{1,l} \ldots \exists p_{N,l}(\mathcal{V}_l \wedge \mathsf{t}(\varphi))
\end{aligned}$$

To facilitate translating formulas of the form $x_{l_1} = x_{l_2}$, the clause for $\mathsf{t}(\exists x_l \varphi)$

provides that the values of the free variables of $\exists x_l \varphi$ are excluded from the range of x_l by treating the cases of $v(x_l)$ being one of these values separately.

The translation of formulas of the form $\mathsf{K}_{x_1,\ldots,x_m}\varphi$ requires us to write a description of $C_E(a,k)$, $k < \omega$, for an arbitrary execution E and a knower a such that $C_E(a,0) = \{v(x_1),\ldots,v(x_m)\}$ in our propositional temporal language. We do this by introducing the propositional variables o_j, $j = 1,\ldots,N$. Just like the variables $p_{j,l}$, o_j have only bound occurrences in the translations of *EPCTL** sentences. Assuming that the considered execution E is written in the form (6), the intended meaning of o_j in the translation of $\mathsf{K}_a \ldots$ at step k is $\sigma_k(y_j) \in C_E(a,k)$. Next we construct an *LTL* formula to express that o_j, $j = 1,\ldots,N$, behave according to the defining properties of $C_E(a,k)$, $k < \omega$, with respect to the adopted way of propositional description of executions E.

Consider an individual variable x_l such that $v(x_l) \in C_E(a,0)$ and let $k < \omega$. Then the satisfaction of

$$I_{j,l} \rightleftharpoons F_{j,l} \Rightarrow (o_j \mathsf{Sch}_j \wedge o_j) \wedge o_j \wedge \bigcirc(o_j \mathsf{Wch}_j)$$

at step k means that if $k \in L_E(v(x_l))$ and $v(x_l) = \sigma_k(y_j)$, then a taps communication over channel $\sigma_k(y_j)$ throughout its extent $L_E(\sigma_k(y_j)) = L_E(v(x_l))$. We put

$$I_L \rightleftharpoons \bigwedge_{j=1}^{N} \bigwedge_{l \in L} \Box I_{j,l} \text{ for } L \subseteq \{1,\ldots,M\}.$$

The satisfaction of I_L at step 0 means that a taps communication over the channels denoted by x_l, $l \in L$, throughout their extents.

To express the definition (4) of $C_E(a,k+1)$ in terms of $C_E(a,k)$, we use the formula

$$C \rightleftharpoons \Box \bigwedge_h (\bigcirc o_h \Leftrightarrow (\neg \bigcirc \mathsf{ch}_h \wedge o_h) \vee \bigvee_j o_j \wedge \mathsf{comm}_{j,h}). \tag{9}$$

The satisfaction of C at step 0, means that communicating a channel name $\sigma_k(y_h)$ over an observed channel $\sigma_k(y_j)$ at an arbitrary step k makes communication over $\sigma_k(y_h)$ observable from step $k+1$ on and for the rest of the extent of $\sigma_k(y_h)$, that is, until eventually a step $k' > k$ is reached such that $\sigma_{k'}(y_h) \neq \sigma_k(y_h)$, which is indicated by ch_h. Let O_L be the formula $\diamondsuit(\mathsf{I} \wedge I_L \wedge C)$. O_L states that o_j holds at step k *iff* $\sigma_k(y_j) \in C_E(a,k)$ for all $k < \omega$ and $j \in L$.

Expressing $\mathsf{K}_{x_1,\ldots,x_m}$ furthermore requires reference to executions E' which exhibit the same sequence of observable actions as the actual execution E. To this end we introduce an extra copy $\mathbf{L}' = \{q'_1,\ldots,q'_K\}$ of the vocabulary $\mathbf{L}$ of our Kripke model M, whose paths we described using the formula $\mathcal{E}$. We write x' for the boolean combination $[q'_i/q_i : i = 1,\ldots,K]\mathsf{x}$, $\mathsf{x} = \mathsf{tau}, \mathsf{busy}_j, \mathsf{comm}_{j_1,j_2}, \mathsf{ch}_j$. Similarly we assume additional sets $p'_{j,l}$, o'_j, $j = 1,\ldots,N$, $l = 1,\ldots,M$, of the variables $p_{j,l}$ and o_j, to describe the extents of the values of individual variables and channel observability in E', and write I'_L, C', etc. for the variants of I_L, C, etc., written in the primed vocabulary.

Let the substitutions involved in writing E' in the form (6) be σ'_k, $k < \omega$. According to our encoding, observing the same actions in E and E' means that if o_j and $\mathsf{comm}_{j,h}$, hold at some step k, then $o_{j'}$ and $\mathsf{comm}'_{j',h'}$ hold for some j', h' such that $\sigma_k(y_j) = \sigma'_k(y_{j'})$ and $\sigma_k(y_h) = \sigma'_k(y_{h'})$. To express the latter identities, we introduce the atomic propositions $e_{j,j'}$, $j, j' = 1, \ldots, N$. The intended meaning of $e_{j,j'}$ at step k is that $\sigma_k(y_j) = \sigma'_k(y_{j'}) \neq *$, that is, $k \in L_E(n) \cap L_{E'}(n)$ where $n = \sigma_k(y_j) = \sigma'_k(y_{j'})$.

The valuation of $e_{j,j'}$, $j, j' = 1, \ldots, N$, along a path describes correctly a possible overlap of the extents $L_E(n)$ and $L_{E'}(n)$ of some name n in a pair of executions E and E', iff it has the properties which are expressed by the following *LTL* formulas

$$e_{j,j'} \Rightarrow \mathsf{busy}_j \wedge \mathsf{busy}'_{j'} \wedge \bigwedge_{h \neq j} \neg e_{h,j'} \wedge \bigwedge_{h' \neq j'} \neg e_{j,h'}$$

$$e_{j,j'} \Rightarrow \bigcirc \left(e_{j,j'} \wedge \neg \mathsf{ch}_j \wedge \neg \mathsf{ch}'_{j'} \mathsf{W} \begin{array}{l} \mathsf{ch}_j \wedge \left(\bigwedge_h \neg e_{h,j'} \wedge \neg \mathsf{ch}'_{j'} \mathsf{W} \mathsf{ch}'_{j'} \right) \vee \\ \mathsf{ch}'_{j'} \wedge \left(\bigwedge_h \neg e_{j,h} \wedge \neg \mathsf{ch}_j \mathsf{W} \mathsf{ch}_j \right) \end{array} \right)$$

$$e_{j,j'} \Rightarrow \left(e_{j,j'} \wedge \neg \mathsf{ch}_j \wedge \neg \mathsf{ch}'_{j'} \mathsf{V} \begin{array}{l} \mathsf{ch}_j \wedge \ominus \left(\bigwedge_h \neg e_{h,j'} \wedge \neg \mathsf{ch}'_{j'} \mathsf{V} \mathsf{ch}'_{j'} \right) \vee \\ \mathsf{ch}'_{j'} \wedge \ominus \left(\bigwedge_h \neg e_{j,h} \wedge \neg \mathsf{ch}_j \mathsf{V} \mathsf{ch}_j \right) \end{array} \right)$$

At step k, the first formula states that $\sigma_k(y_j) = \sigma'_k(y_{j'}) \neq *$ can hold for at most one pair j, j'. The second and the third formulas state that $\sigma_k(y_j) = \sigma'_k(y_{j'}) = n$ at step k implies $\sigma_{k'}(y_j) = \sigma'_{k'}(y_{j'})$ for all $k' \in L_E(n) \cap L_{E'}(n)$, $\sigma_{k'}(y_j) \neq \sigma'_{k'}(y_h)$ for all h and $k' \in L_E(n) \backslash L_{E'}(n)$, and $\sigma'_{k'}(y_{j'}) \neq \sigma_{k'}(y_h)$ for all h and $k' \in L_{E'}(n) \backslash L_E(n)$. Let $\mathcal{N}_{j,j'}$ be the conjunction of these formulas. We denote the formula $\Diamond(\mathsf{I} \wedge \forall \Box \bigwedge_{j,j'} \mathcal{N}_{j,j'})$ by $\mathcal{N}$.

Using the variables $e_{j,j'}$ we can express that E and E' have the same observable communication by the formulas

$$o_j \Rightarrow \bigwedge_h (e_{j,h} \Rightarrow o'_h), \qquad o'_{j'} \Rightarrow \bigwedge_h (e_{k,j'} \Rightarrow o_h) \tag{10}$$

$$e_{j,j'} \Rightarrow (p_{j,l} \Leftrightarrow p'_{j',l}) \tag{11}$$

$$o_j \wedge \mathsf{comm}_{j,h} \Rightarrow \bigvee_{j',h'} (e_{j,j'} \wedge e_{h,h'} \wedge \mathsf{comm}'_{j',h'}) \tag{12}$$

$$o'_{j'} \wedge \mathsf{comm}'_{j',h'} \Rightarrow \bigvee_{j,h} (e_{j,j'} \wedge e_{h,h'} \wedge \mathsf{comm}_{j,h}). \tag{13}$$

The formulas (10) state that $C_E(a,k) = C_{E'}(a,k)$ for the reference step k. The formula (11) states that the account of the valuation of individual variables given by $p_{j,l}$ and $p'_{j',l}$ is consistent with the identities between in E and E' as described using $e_{j,j'}$. The formulas (12) and (13) state that the actions on observable channels in the two executions are identical. We denote the conjunction of (10)-(13) by $\mathcal{S}_{j,j',h,h'}$. We denote $\boxminus \bigwedge_{j,j',h,h'} \mathcal{S}_{j,j',h,h'}$ by $\mathcal{S}$. The satisfaction of $\mathcal{S}$ at step k means that $E \sim_{a,k} E'$

holds, provided that executions E and E' correspond to the satisfying path, that is, provided that busy_j, ch_j, tau, comm_{j_1,j_2}, $\mathsf{busy}'_{j'}$, $\mathsf{ch}'_{j'}$, tau and $\mathsf{comm}'_{j'_1,j'_2}$ correctly describe E and E', respectively, $p_{j,l}$, $p'_{j',l}$ and $e_{j,j'}$ correctly describe the identities between the names involved in E and E', and the values of the individual variables x_l, and, finally, o_j and o'_j, correctly describe the observability of channels. This condition is expressed by the conjunction

$$\mathcal{N} \wedge \mathcal{E} \wedge \bigwedge_{x_n \in FV(\varphi)} \mathcal{V}_n \wedge O_{\{1,\ldots,m\}} \wedge \left(\mathcal{E}' \wedge \bigwedge_{x_n \in FV(\varphi)} \mathcal{V}'_n \wedge O'_{\{1,\ldots,m\}} \right)$$

The subscripts written with i and j, and also l as the main symbol above range over $\{1,\ldots,N\}$ and $\{1,\ldots,M\}$, respectively.

Now we are ready to write a translation clause for $\mathsf{K}_{x_1,\ldots,x_m}\varphi$. (The initially observable channels are chosen to be values of the first m individual variables $x_1,\ldots,x_m$ for the sake of simplicity.) $\mathsf{K}_{x_1,\ldots,x_m}\varphi$ translates into

$$\begin{array}{ll} \forall q'_1 \ldots \forall q'_1 & \\ \forall p'_{1,1} \ldots \forall p'_{1,M} \ldots \forall p'_{N,1} \ldots \forall p'_{N,M} & \left(\begin{array}{l} \mathcal{N} \wedge \mathcal{S} \wedge \mathcal{E}' \wedge \bigwedge\limits_{x_n \in FV(\varphi)} \mathcal{V}'_n \wedge \\ O_{\{1,\ldots,m\}} \wedge O'_{\{1,\ldots,m\}} \Rightarrow \mathsf{t}(\varphi)' \end{array} \right) \\ \forall o_1 \ldots \forall o_N \forall o'_1 \ldots \forall o'_N & \\ \forall e_{1,1} \ldots \forall e_{1,N} \ldots \forall e_{N,1} \ldots \forall e_{N,N} & \end{array}$$

The quantifier prefix of $\mathsf{t}(\mathsf{K}_{x_1,\ldots,x_m}\varphi)$ provides fresh sets of variables $q'_1,\ldots,q'_K$ to enable the description of E', $p'_{j',l'}$ to describe the identities between the values of the individual variables and the names involved in E', o_j and $o'_{j'}$ to mark the observability of channels in E and E', respectively, and a set of variables $e_{j,j'}$ to express whatever identities hold between the names occurring in E and E' during their various extents. The conditions on these variables which actually force their truth values to give a consistent account of E', the way individual variables refer to names in E', the observability of channels in both executions, the identities between names occurring in E and E', and the fact that $E \sim_a E'$ for a knower a who can initially observe the channels $v(x_1),\ldots,v(x_m)$ are expressed in the conjunction on the left of $\Rightarrow$ in the matrix of the formula by $\mathcal{E}'$, $\bigwedge\limits_{x_n \in FV(\varphi)} \mathcal{V}'_n$, $O_{\{1,\ldots,m\}}$, $O'_{\{1,\ldots,m\}}$, $\mathcal{N}$, and $\mathcal{S}$, respectively. On the whole, the translation states that if a cannot tell apart some E' from the actual execution E, then the encoding of E' satisfies $\mathsf{t}(\varphi)$ as well, which is the defining condition for the satisfaction of $\mathsf{K}_{x_1,\ldots,x_m}\varphi$. The free propositional variables of $\mathsf{t}(\mathsf{K}_{x_1,\ldots,x_m}\varphi)$ are $q_1,\ldots,q_K$, and $p_{j,l}$, $j = 1,\ldots,N$, $x_l \in FV(\varphi)$, which describe the actual execution E and the identities between the names occurring in E and the values of the (free) variables of φ, provided that their truth values satisfy $\mathcal{E}$ and the relevant $\mathcal{V}_l$, respectively.

The correctness of our translation can be formuated as follows:

Theorem 3.1 *Given a π-process P^0 and an EPCTL* sentence φ in the respective predicate vocabulary, $\mathcal{T}(P^0), v, \langle P^0, \emptyset\rangle \models \varphi$ iff $\models_{QCTL^*} \mathcal{E} \Rightarrow \mathsf{t}(\varphi)$.*

The proof can be obtained by following the detailed explanation of the meaning of the formulas used to define the various clauses for $\mathsf{t}(.)$ above.

4 Unsolvability of model-checking with external communication

Model-checking is not recursively enumerable for finite control π-processes with external communication even for the *LTL* subset of $EPCTL^*$, without the epistemic modality. To prove this, we define a class of behaviours in which the environment e acts as unbounded storage by an *LTL* formula. Let I and O be binary predicate symbols which denote input from and output to e, respectively, just like the predicate symbol C about internal communication. We intend to state that whenever e receives two names x and y in a row along a dedicated channel *cons*, it "registers" the pair $\langle x, y\rangle$ under some name z and, from that step onwards, whenever given z along the dedicated channels *car* (*cdr*), e sends back x (y) along *reply*. A formula constraining e to behave this way can be written as follows. The formula

$$\mathit{silence} \rightleftharpoons \forall x\big(\wedge_{c\in\{\mathit{cons},\mathit{car},\mathit{cdr},\mathit{reply}\}}\neg I(c,x)\wedge\neg O(c,x)\big).$$

states that the latest action was not communication with e. Let

$$\Diamond_s\varphi \rightleftharpoons (\mathit{silence}\mathsf{U}\varphi) \text{ and } \Diamondblack_s\varphi \rightleftharpoons (\mathit{silence}\mathsf{S}\varphi);$$
$$R \rightleftharpoons \boxminus\mathit{silence}\vee\Diamondblack_s\exists x I(\mathit{reply},x).$$

R states that the latest communication with e, if any, was a reply. Then

$$\Diamondblack_s(O(\mathit{cons},y)\wedge\ominus\Diamondblack_s(O(\mathit{cons},x)\wedge\ominus R))\Rightarrow$$
$$\exists z\Diamond_s(I(\mathit{reply},z)\wedge\bigcirc\forall t\Box(O(\mathit{car},z)\wedge\bigcirc\Diamond_s(I(\mathit{reply},t)\Rightarrow t=x)))$$

states that e is bound to return x whenever asked to retrieve the first member of the pair $\langle x, y\rangle$ previously registered as z. Similar formulas can be written to express retrieving y, and registering pairs. We leave it to the reader to realise that, with e assumed to behave this way, a finite control process P_M can be constructed to simulate the working of any given Turing machine M, with the parts of M's tape on the left and on the right of M's current position represented as two lists built of pairs, which can be stored by e in the above fashion. This entails that the non-halting problem for Turing machines M reduces to the model-checking problem for processes of the form P_M against the conjunction of the formulas which describe the working of e as storage and a formula which states the non-termination of M.

The same plan can be used to show that the problem of model-checking finite-control processes which communicate with a *finite memory* environment for predicate *LTL* properties, that is, the problem of whether there exists a finite-control E such that the runs of $P \mid E$ for a given P have a given property written in the *LTL* subset of our $EPCTL^*$, is recursively enumerable but still undecidable, as long as E

is unrestricted. This can be realised by choosing P to range over the processes P_M which simulate Turing machines M as above, and the property in question to be *M terminates and E behaves as storage in the above way until M terminates.* By restricting M to be deterministic, $P_M|E$ can be chosen to have just one run. For terminating M, the unique run of $P_M \mid E$ will satisfy the above property for any E which is big enough to serve as storage throughout the terminating run of the simulated M.

Concluding remarks

We have examined model checking of finite control π-calculus processes against formulas in an epistemic extension of predicate CTL^* with perfect recall. Since model checking is undecidable for open π-calculus processes even for LTL, we instead address closed process terms and tapping internal communication across a distinguished set of channels. This constrains the storage capacity of processes sufficiently to render model checking decidable.

Model checking the π-calculus has been considered by several authors, but so far only in branching time settings. Dam [5] obtained a first decidability result for a predicate extension of modal μ-calculus. This result has been improved upon in [7,17]. The latter work has been adapted to the stochastic π-calculus [13]. Recent applications of π-calculus and its dialects to security protocol verification mostly appeal to Dolev-Yao type knowledge extraction. An exception is [3], where the use of epistemic reasoning in the context of π-calculus is suggested. An epistemically flavoured extension of modal logic applied to CCS, a precursor of the π-calculus, is proposed in [11].

We leave three main questions open for future investigation. First, we have not explored the practical implications of the closed system modelling approach suggested in this paper, and whether it can offer new approaches to specification and verification, for instance along the lines suggested by [3]. Second, the model checking algorithm presented here is non-elementary and needs to be improved in order to become practically useful. It remains to be seen if existing approaches to model checking of epistemic logics [10,15] can be extended. Third, it is of interest to extend the results presented here to capture also strategic ability, for instance along the lines of ATL [2].

Acknowledgement

Dimitar Guelev worked on the topic of this paper during a research visit to KTH in September, 2009, which was partially supported by the ACCESS Linnaeus Excellence Centre, funded by the Swedish National Research Council. The work was also partly supported by Bulgarian National Science Fund Grant ID-09-112. D. Guelev is grateful to Mark Ryan for some comments on a draft version of the paper.

References

[1] Abadi, M. and C. Fournet, *Mobile values, new names, and secure communication*, in: *POPL* (2001), pp. 104–115.

[2] Alur, R., T. A. Henzinger and O. Kupferman, *Alternating-time temporal logic*, J. ACM **49** (2002), pp. 672–713.

[3] Chadha, R., S. Delaune and S. Kremer, *Epistemic Logic for the Applied Pi Calculus*, in: *FMOODS/FORTE*, LNCS **5522** (2009), pp. 182–197.

[4] Cohen, M. and M. Dam, *A complete axiomatization of knowledge and cryptography*, in: *LICS* (2007), pp. 77–88.

[5] Dam, M., *Model Checking Mobile Processes*, Information and Computation **129** (1996), pp. 35–51.

[6] Fagin, R., J. Halpern, Y. Moses and M. Vardi, "Reasoning about Knowledge," MIT Press, 1995.

[7] Franzen, T., *A Theorem-Proving Approach to Deciding properties of Finite-Control Agents* (1996).

[8] French, T., *Decidability of quantifed propositional branching time logics*, in: *AI 2001*, LNCS **2256** (2001), pp. 165–176.

[9] French, T., "Bisimulation Quantifiers for Modal Logics," Ph.d. thesis, The University of Western Australia (2006).

[10] Gammie, P. and R. van der Meyden, *MCK: Model Checking the Logic of Knowledge*, in: *CAV*, 2004, pp. 479–483.

[11] Mardare, R., *Observing Distributed Computation. A Dynamic-Epistemic Approach*, in: *CALCO*, LNCS **4624** (2007), pp. 379–393.

[12] Milner, R., J. Parrow and D. Walker, *A Calculus of Mobile Processes, I*, Information and Computation **100** (1992), pp. 1–40.

[13] Norman, G., C. Palamidessi, D. Parker and P. Wu, *Model checking the probabilistic pi-calculus*, in: *QEST* (2007), pp. 169–178.

[14] Parrow, J., *An introduction to the π-calculus*, in: J. Bergstra, A. Ponse and S. Smolka, editors, *Handbook of Process Algebra*, Elsevier Science Inc., New York, NY, USA, 2001 pp. 479–544.

[15] Raimondi, F. and A. Lomuscio, *Automatic Verification of Multi-agent Systems by Model Checking via Ordered Binary Decision Diagrams*, Journal of Applied Logic **5** (2007), pp. 235 – 251.

[16] Sangiorgi, D. and D. Walker, "The π-calculus: a Theory of Mobile Processes," CUP, 2001.

[17] Yang, P., C. R. Ramakrishnan and S. A. Smolka, *A logical encoding of the pi-calculus: model checking mobile processes using tabled resolution*, STTT **6** (2004), pp. 38–66.