When Ambients Cannot be Opened

Iovka Boneva, Jean-Marc Talbot

When Ambients Cannot be Opened *

Iovka Boneva and Jean-Marc Talbot
{boneva,talbot}@lifl.fr

Laboratoire d'Informatique Fondamentale de Lille, France.

Abstract. We investigate expressiveness of a fragment of the ambient calculus, a formalism for describing distributed and mobile computations. More precisely, we study expressiveness of the pure and public ambient calculus from which the capability `open` has been removed, in terms of the reachability problem of the reduction relation. Surprisingly, we show that even for this very restricted fragment, the reachability problem is not decidable. At a second step, for a slightly weaker reduction relation, we prove that reachability can be decided by reducing this problem to markings reachability for Petri nets. Finally, we show that the name-convergence problem as well as the model-checking problem turn out to be undecidable for both the original and the weaker reduction relation.

1 Introduction

The ambient calculus [5] is a formalism for describing distributed and mobile computation in terms of *ambients*, named collections of running processes and nested sub-ambients. A state of computation has a tree structure induced by ambient nesting. Mobility is represented by re-arrangement of this tree (an ambient may move inside or outside other ambients) or by deletion of a part of this tree (a process may dissolve the boundary of some ambient, revealing its contents). Mobility is ruled by the capabilities `in`, `out`, `open` owned by the ambients. The ambient calculus also inherits replication, name restriction and asynchronous communication from the π-calculus [16].

The ambient calculus is a very expressive formalism. It has been proved Turing-complete in [5] by encoding Turing-machine computations. This encoding uses both mobility features from the calculus as well as name restriction. However, several variants of the ambient calculus have been proposed so far [14, 2, 19] by adding and/or retracting features from the original calculus. In [14], the safe ambient calculus introduces some co-capabilities. They are used as an agreement on mobility between the moving ambient (executing a capability) and the ambient where it will move to (executing the corresponding co-capability). The boxed ambient calculus is another variant [2]; in this calculus, the possibility to dissolve boundary of ambients has disappeared and is replaced by a more sophisticated communication mechanism.

Studying precise expressiveness of these different variants of the ambient calculus is crucial as it may separate necessary features from redundant ones and it may also help to design or improve algorithms to verify [18, 7, 6] or analyze [11, 9] programs written in these ambient formalisms.

* The authors are grateful to S. Tison and Y. Roos for fruitful discussions and thank the anonymous referees for valuable comments. This work is supported by an ATIP grant from CNRS.

Some works aimed already to study expressiveness of ambient calculus: in [20], it is shown that the π-calculus, a formalism based only on name communication, can be simulated in the communication-free safe ambient calculus. In [7], the pure and public ambient calculus (an ambient calculus in which communication and name restriction are omitted) is considered and proved to be still very powerful: for this restricted fragment, the reachability problem (*i.e.* given two processes P and Q, can the process P evolve into the process Q ?) can not be decided. Recently, in two different works [13] and [3], it has been established that this fragment is actually Turing-complete. In [3], the authors also showed that the ambient calculus limited to `open` capabilities and name restriction is Turing-complete as it can simulate counters machines computations [17]. The name restriction is needed there as if omitted, divergence for reductions of processes can be decided. In this latter paper, the following question is raised: what is the expressiveness power of the "dual" calculus, a calculus in which the `open` capability is dropped whereas the `in`, `out` capabilities are preserved ?

In this paper, we investigate expressiveness of pure and public mobile ambients without the `open` capability. Hence, the reduction of a process is limited to the rearrangement of its tree structure. To be more precise, we study the reachability problem for such ambient processes. We show that for this calculus reachability for the reduction relation between two processes can not be decided. To prove this result, we use a non-trivial reduction to the acceptance problem of two-counters machines [17]. We figured out that the major source of undecidability for this fragment comes from the definition of replication as part of the structural congruence relation (the structural congruence aims to identify equivalent processes having different syntactic representations). Indeed, we show that if this definition of replication is presented as an (oriented) reduction rule then the reachability for the reduction relation between two processes becomes decidable. We prove this statement by reducing this problem to the reachability problem of markings in Petri nets [15]. Finally, we investigate two problems related to reachability. The first problem is the name-convergence problem [10]: a process converges to some name n if this process can be reduced to some process having an ambient named n at its top-level. We show that this problem is undecidable however the definition of replication is presented. The second problem is the model-checking problem against the ambient logic [4]. It is easy to show that the name-convergence problem can be reduced to an instance of the model-checking problem. Thus, this latter is undecidable as well.

The paper is organized as follows: in Section 2, we give definitions for the ambient calculus we consider here. Section 3 is devoted to the reachability problem for this fragment. We give there a negative result: this problem is undecidable. In Section 4, we consider a weak calculus based on a different reduction relation; we show that for this particular case the reachability problem becomes decidable. Finally, in Section 5, we consider other problems such as model-checking and name-convergence. We show that for the two kinds of reduction we considered those problems are not decidable.

2 Definitions

We present in this section the fragment of the ambient calculus we consider all along this paper. This fragment corresponds to the ambient calculus defined in [5] for which

both name restriction, communication and the `open` capability have been dropped. We call this fragment `in`/`out` ambient calculus.

We assume countably many *names* ranging over by $n, m, a, b, c, \ldots$. For any name n, we consider *capabilities* α of the form `in` n and `out` n. The following table defines the syntax of *processes* of our calculus.

Processes:

$P, Q, R ::=$	processes		
$\mathbf{0}$	inactivity	$P \mid Q$	composition
$n[P]$	ambient	$\alpha.P$	action prefix
$!P$	replication		

The semantics of our calculus is given by two relations. The *reduction relation* $P \rightarrow Q$ describes the evolution of processes over time. We write $\rightarrow^*$ for the reflexive and transitive closure of $\rightarrow$. The *structural congruence* relation $P \equiv Q$ relates different syntactic representations of the same process; it is used to define the reduction relation.

The structural congruence is defined as the least relation over processes satisfying the axioms from the table below:

Structural Congruence $P \equiv Q$:

$P \equiv P$	(Str Refl)	$P \mid \mathbf{0} \equiv P$	(Str Par Zero)
$P \equiv Q \Rightarrow Q \equiv P$	(Str Symm)	$P \mid Q \equiv Q \mid P$	(Str Par Comm)
$P \equiv Q, Q \equiv R \Rightarrow P \equiv R$	(Str Trans)	$(P \mid Q) \mid R \equiv P \mid (Q \mid R)$	(Str Par Assoc)
$P \equiv Q \Rightarrow P \mid R \equiv Q \mid R$	(Str Par)	$!P \equiv !P \mid P$	(Str Repl Copy)
$P \equiv Q \Rightarrow n[P] \equiv n[Q]$	(Str Amb)	$!\mathbf{0} \equiv \mathbf{0}$	(Str Repl Zero)
$P \equiv Q \Rightarrow \alpha.P \equiv \alpha.Q$	(Str Action)	$!!P \equiv !P$	(Str Repl Repl)
$P \equiv Q \Rightarrow !P \equiv !Q$	(Str Repl)	$!(P \mid Q) \equiv !P \mid !Q$	(Str Repl Par)

The first column specifies that $\equiv$ is a congruence relation over processes. The second one specifies properties of the replication and parallel operators: in particular, it states that the parallel operator is associative-commutative and has $\mathbf{0}$ as neutral element.

The reduction relation is defined as the least relation over processes satisfying the following set of axioms:

Reduction: $P \rightarrow Q$

$n[\texttt{in}\ m.P \mid Q] \mid m[R] \rightarrow m[n[P \mid Q] \mid R]$	(Red In)
$m[n[\texttt{out}\ m.P \mid Q] \mid R] \rightarrow n[P \mid Q] \mid m[R]$	(Red Out)
$P \rightarrow Q \Rightarrow P \mid R \rightarrow Q \mid R$	(Red Par)
$P \rightarrow Q \Rightarrow n[P] \rightarrow n[Q]$	(Red Amb)
$P' \equiv P, P \rightarrow Q, Q \equiv Q' \Rightarrow P' \rightarrow Q'$	(Red $\equiv$)

When writing processes, we may omit irrelevant occurrences of the inactive process $\mathbf{0}$. For instance, we may write $n[]$ for $n[\mathbf{0}]$ and `in` a.`out` b for `in` a.`out` b.$\mathbf{0}$.

3 The reachability problem for `in`/`out` ambient calculus

In this section we investigate the reachability problem for `in`/`out` ambient calculus: "Given two processes P, Q, does $P \rightarrow^* Q$ hold ?". We show that, despite the small

fragment of the ambient calculus we consider, this problem is undecidable by defining a reduction to the acceptance problem of two-counters machine [17].

A *two-counters machine* $\mathcal{M}$ is given by a four tuple $(\mathcal{Q}, q_i, q_f, \Delta)$ where $\mathcal{Q}$ is a finite set of states, $q_i \in \mathcal{Q}$ is the initial state, $q_f \in \mathcal{Q}$ is the final state; Δ is a finite subset of $(\mathcal{Q} \times \{+, -, =\} \times \{0, 1\} \times \mathcal{Q})$ called the *transition relation*. A *configuration* of the machine $\mathcal{M}$ is given by a triple (q, c_0, c_1) belonging to the set $(\mathcal{Q}, \mathbb{N}, \mathbb{N})$ (where $\mathbb{N}$ is the set of natural numbers); c_0, c_1 are the two counters. The transition relation Δ defines a *step relation* $\vdash_{\mathcal{M}}$ over configurations as follows: $(q, c_0, c_1) \vdash_{\mathcal{M}} (q', c_0', c_1')$ iff one of the three statements is true for $i, j \in \{0, 1\}$ and $i \neq j$: (**i**) $(q, =, i, q')$ in Δ, $c_i = c_i' = 0$ and $c_j = c_j'$, (**ii**) $(q, +, i, q')$ in Δ, $c_i' = c_i + 1$ and $c_j = c_j'$, (**iii**) $(q, -, i, q')$ in Δ, $c_i > 0$, $c_i' = c_i - 1$ and $c_j = c_j'$.

Let $\vdash_{\mathcal{M}}^*$ be the reflexive and transitive closure of $\vdash_{\mathcal{M}}$. A two-counters machine *accepts* a natural v if $(q_i, v, 0) \vdash_{\mathcal{M}}^* (q_f, 0, 0)$.

Theorem 1. *[17] For an arbitrary two-counters machine $\mathcal{M}$ and an arbitrary natural v, it is undecidable to test whether $\mathcal{M}$ accepts v.*

We express the acceptance problem of a natural v by a machine $\mathcal{M}$ in terms of an instance of the reachability problem in the `in`/`out` ambient calculus. We encode within a process $[\![(q, v_0, v_1)]\!]$ the configuration of the machine (q, v_0, v_1) (the current state and the values for the two counters) and the transition relation Δ of the machine. The step relation of the machine is then simulated by the reduction of this process. It should be noticed that not all the different reductions that could be performed by the process indeed participate to the step relation; the process may engage some wrong reduction steps. However, we show that in this case, the process either goes stuck in a form that does not correspond to some valid representation of the machine or carries on some reduction steps but without any possibility to resume into an encoding of the two-counters machine. Let us now describe the process $[\![(q, v_0, v_1)]\!]$: we assume for any state q occurring in Δ, two ambient names q and q_t: q represents the state of the machine and q_t is used to denote a possible transition of the machine being in the state q. The process $[\![(q, v_0, v_1)]\!]$ is defined as

$$q[\texttt{in}\ q_t] \mid c_0[V(v_0) \mid !k[\texttt{out}\ c_0]] \mid c_1[V(v_1) \mid !k[\texttt{out}\ c_1]] \mid [\![\Delta]\!] \mid P_G$$

The ambient $q[\texttt{in}\, q_t]$ represents the current state of the machine; the ambients c_0 and c_1 represent the counters with their respective values $V(v_0)$ and $V(v_1)$. The parametrized process $V(v)$ encodes the value v recursively as: (i) $V(0) = n[!I \mid !D \mid \texttt{in}\ k \mid a[\mathbf{0}]]$ and (ii) $V(v+1) = n[!I \mid !D \mid \texttt{in}\ k \mid b[\mathbf{0}] \mid V(v)]$. Intuitively, the value v of the counter is given by the number of ambients n in the process $V(v)$ minus 1. The two processes I and D are defined as $I = \texttt{in}\, i.\texttt{in}\, n.\mathbf{0}$, $D = \texttt{in}\, z.\texttt{in}\, z'.\texttt{out}\, z'.\texttt{out}\, n.\mathbf{0}$: the process I is used to increment the counter and D to decrement it.

The process $[\![\Delta]\!]$ represents the transition rules of the machine and is defined as the parallel composition of the replicated processes encoding each transition rule. Formally, we have inductively $[\![\varnothing]\!] = \mathbf{0}$ and $[\![\Delta \cup \{(q, s, j, q')\}]\!] = [\![\Delta \setminus \{(q, s, j, q')\}]\!] \mid ![\![(q, s, j, q')]\!]$. For each kind of transition rules:

- $[\![(q, =, j, q')]\!] = q_t[q'[\beta_j^q.(\texttt{in}\, n.\texttt{in}\, a.\texttt{out}\, a.\texttt{out}\, n.\texttt{out}\, c_j.\texttt{in}\, q_t')]]$

– $[\![(q,+,j,q')]\!] = q_t[i[\beta_j^q.N_{q'} \mid \texttt{in}\, q'.\texttt{out}\, q'.\texttt{out}\, c_j]]$ with $N_{q'} = n[\texttt{out}\, i.(!I \mid !D \mid \texttt{in}\, k \mid b[\mathbf{0}] \mid q'[\delta_{q'}.\texttt{in}\, q_t'])]$ and $\delta_{q'} = \texttt{in}\, n.\texttt{out}\, n.\texttt{out}\, n.\texttt{in}\, i.\texttt{out}\, i$.

– $[\![(q,-,j,q')]\!] = q_t[d[\beta_j^q.\texttt{in}\, n.\texttt{in}\, b.\texttt{out}\, b.Z_q']]$ with $Z_{q'} = z[\texttt{out}\, d.z'[\texttt{out}\, z.q'[\texttt{out}\, z'.\texttt{out}\, n.\texttt{out}\, k.\texttt{in}\, q_t']]]$

where β_j^q is defined as the sequence of capabilities $\texttt{in}\, q.\texttt{out}\, q.\texttt{out}\, q_t.\texttt{in}\, c_j$.

Finally, the process P_G plays the role of some garbage collector (using that $!P \mid P$ is structurally congruent to $!P$): assuming that $q^1, \ldots, q^l$ are the states of the two-counters machine, the process P_G is defined as

$$!k[\mathbf{0}] \mid !i[\mathbf{0}] \mid !k[n[!I \mid !D \mid d[\mathbf{0}] \mid z[\mathbf{0}] \mid z'[\mathbf{0}] \mid b[\mathbf{0}]]] \mid !q_t^1[q^1[\mathbf{0}]] \mid \ldots \mid !q_t^l[q^l[\mathbf{0}]]$$

Note that at any step of the computation, because of the two subprocesses $!k[\texttt{out}\, c_0]$ and $!k[\texttt{out}\, c_1]$ contained respectively in the ambients c_0 and c_1, the process can always perform a reduction step: a copy of $k[\texttt{out}\, c_j]$ can "jump" outside of the counter c_j. However, the resulting process $k[\mathbf{0}]$ is simply garbage-collected by the process P_G. Hence, the process would simply reduce into itself; therefore, we will not take any longer into account these irrelevant reduction steps.

Let us now describe how the process $[\![(q, v_0, v_1)]\!]$ may evolve into another process $[\![(q', v_0', v_1')]\!]$ according to the transition rules of the two-counters machine. First, the ambient $q[\texttt{in}\, q_t]$ reduces with a sibling ambient q_t. Note that a misleading reduction with an ambient $q_t[q[\mathbf{0}]]$ from the process P_G may happen. If so, the computation is stuck. If there exists a transition from the state q in the two-counter machine, then an alternative reduction could occur with an ambient q_t provided by Δ leading to

$$q_t[q[\mathbf{0}] \mid \eta[\beta_j^q.\ldots \mid \ldots]] \mid c_0[\ldots] \mid c_1[\ldots] \mid [\![\Delta]\!] \mid P_G \qquad \text{with } \eta \in \{i, d, q'\}$$

The sequence of capabilities β_j^q allows the transition which has the "control" (*i.e.* q_t contains $q[\mathbf{0}]$) to provide "material" (represented as the ambient η) to treat the counter addressed by this transition. Once, β_j^q is executed we obtain (assuming the transition addresses the counter c_0) $c_0[\eta[\ldots] \mid \ldots] \mid c_1[\ldots] \mid [\![\Delta]\!] \mid P_G$ with $\eta \in \{i, d, q'\}$.

Note that $q_t[q[\mathbf{0}]]$ remaining at the top-level is garbage collected by P_G. Now, the reductions differ according to the kind of transition that was chosen (assuming the transition addresses the counter c_0, things being similar for c_1):

• **for** $(\mathbf{q}, =, \mathbf{0}, \mathbf{q}')$**:** the ambient $\eta[\ldots]$ is $q'[\texttt{in}\, n.\texttt{in}\, a.\texttt{out}\, a.\texttt{out}\, n.\texttt{out}\, c_0.\texttt{in}\, q_t']$. If the value of c_0 is 0 then c_0 contains an ambient $n[a[\mathbf{0}] \mid \ldots]$. The sequence of capabilities $\texttt{in}\, n.\texttt{in}\, a.\texttt{out}\, a.\texttt{out}\, n.\texttt{out}\, c_0$ can be executed and so, the next configuration is obtained. Note that if $V(v_0)$ is not 0, then the process remains stuck as q can not execute its capability $\texttt{in}\, a$.

• **for** $(\mathbf{q}, +, \mathbf{0}, \mathbf{q}')$**:** the ambient $\eta[\ldots]$ is $i[N_{q'} \mid \texttt{in}\, q'.\texttt{out}\, q'.\texttt{out}\, c_0]]$ and the value $V(v_0)$ of the form $n[!I \mid \ldots]$ is one if its siblings. Reminding that $N_{q'} = n[\texttt{out}\, i.(!I \mid !D \mid \texttt{in}\, k \mid b[\mathbf{0}] \mid q'[\delta_{q'}.\texttt{in}\, q_t'])]$, we can see that $N_{q'}$ contains roughly an ambient n used for incrementing and the successor state q' that will try to check that the incrementing has been done properly.

$V(v_0)$ executes the sequence $I = \texttt{in}\, i.\texttt{in}\, n$ leading to an ambient c_0

$$c_0\left[i\left[\begin{array}{l} n[\texttt{out}\, i.(!I \mid !D \mid \texttt{in}\, k \mid b[\mathbf{0}] \mid q'[\delta_{q'}.\texttt{in}\, q_t']) \mid V(v_0)] \\ \mid \texttt{in}\, q'.\texttt{out}\, q'.\texttt{out}\, c_0 \end{array}\right] \mid !k[\texttt{out}\, c_0]\right]$$

By executing $\texttt{out}\ i$ from the top ambient n, one obtains

$$c_0[i[\texttt{in}\ q'.\texttt{out}\ q'.\texttt{out}\ c_0] \mid n[V(v_0) \mid\ !I \mid\ !D \mid\ \texttt{in}\ k \mid b[\mathbf{0}] \mid q'[\delta_{q'}.\texttt{in}\ q'_t]] \mid\ !k[\texttt{out}\ c_0]]$$

At that point, we have almost $V(v_0+1)$ except the presence of q' in the top ambient n. Then, by using $\delta_{q'} = \texttt{in}\ n.\texttt{out}\ n.\texttt{out}\ n.\texttt{in}\ i.\texttt{out}\ i$, q' notices that it has $V(v_0)$ as a sibling (by executing $\texttt{in}\ n.\texttt{out}\ n$) and goes outside of n.

$$c_0[i[\texttt{in}\ q'.\texttt{out}\ q'.\texttt{out}\ c_0] \mid q'[\texttt{in}\ i.\texttt{out}\ i.\texttt{in}\ q'_t] \mid V(v_0+1) \mid\ !k[\texttt{out}\ c_0]]$$

The ambient i detects it has a sibling q' (by executing $\texttt{in}\ q'.\texttt{out}\ q'$) and the ambient q' enters i yielding $c_0[i[\texttt{out}\ c_0 \mid q'[\texttt{out}\ i.\texttt{in}\ q'_t]] \mid V(v_0+1) \mid\ !k[\texttt{out}\ c_0]]$. Then i goes outside of c_0. So, the process is

$$i[q'[\texttt{out}\ i.\texttt{in}\ q'_t]] \mid c_0[V(v_0+1) \mid \ldots] \mid c_1[V(v_1) \mid \ldots] \mid [\![\Delta]\!] \mid P_G$$

The ambient q' exits i, producing the process $i[\mathbf{0}] \mid q'[\texttt{in}q'_t]$. The process $i[\mathbf{0}]$ is garbage-collected by P_G, and the result indeed corresponds to the process $[\![(q', v_0+1, v_1)]\!]$.

• **for** $(\mathbf{q}, -, \mathbf{0}, \mathbf{q}')$**:** the ambient $\eta[\ldots]$ is $d[\texttt{in}\ n.\texttt{in}\ b.\texttt{out}\ b.Z_{q'}]$. If the value of $V(v_0)$ is strictly positive then it is of the form $n[b[\mathbf{0}] \mid \ldots]$. Then d executes $\texttt{in}\ n.\texttt{in}\ b.\texttt{out}\ b$; the contents of c_0 is then $c_0[n[d[Z_{q'}] \mid V(v_0-1) \mid\ !I \mid\ !D \mid\ \texttt{in}\ k \mid b[\mathbf{0}]] \mid\ !k[\texttt{out}\ c_0]]$.

Note that if $V(v_0)$ is 0, then the process remains stuck as d can not execute its capability $\texttt{in}\ b$. The role of $Z_{q'}$ is interact with $V(v_0-1)$ in order to trigger for this latter the possibility to go outside of its surrounding ambient n. We recall that $Z_{q'} = z[\texttt{out}\ d.z'[\texttt{out}\ z.q'[\texttt{out}\ z'.\texttt{out}\ n.\texttt{out}\ k.\texttt{in}\ q'_t]]]$ and that $V(v_0-1)$ contains at its top-level $D = \texttt{in}\ z.\texttt{in}\ z'.\texttt{out}\ z'.\texttt{out}\ n.\mathbf{0}$. The ambient z from $Z_{q'}$ exits d; then, the ambient $V(v_0-1)$ executes the capabilities $\texttt{in}\ z.\texttt{in}\ z'$ from D and finally, z' leaves z. We reach the following situation for the ambient z':

$$z'[q'[\texttt{out}\ z'.\texttt{out}\ n.\texttt{out}\ k.\texttt{in}\ q'_t] \mid n[\texttt{out}\ z'.\texttt{out}\ n \mid \texttt{in}\ k \mid \ldots]]$$

The ambient n executes the remaining capabilities from D, that is $\texttt{out}\ z'.\texttt{out}\ n$; concurrently, the ambient q' exits z'. The contents of c_0 is then

$$!k[\texttt{out}\ c_0] \mid V(v_0-1) \mid n[\texttt{in}\ k \mid\ !I \mid\ !D \mid b[] \mid d[] \mid z[] \mid z'[] \mid q'[\texttt{out}\ n.\texttt{out}\ k.\texttt{in}\ q'_t]]$$

At that point, the value of the counter has been decremented; the rest of the computation aims to "clean up" the process allowing the computation to carry on. The ambient n moves inside an ambient $k[\texttt{out}\ c_0]$. So, we have in c_0

$$!k[\texttt{out}\ c_0] \mid V(v_0-1) \mid k[\texttt{out}\ c_0 \mid n[!I \mid\ !D \mid b[] \mid d[] \mid z[] \mid z'[] \mid q'[\texttt{out}\ n.\texttt{out}\ k.\texttt{in}\ q'_t]]]$$

In some arbitrary order, the ambient k (containing n) leaves the counter c_0 and q' leaves the ambient n.

$$k[n[!I \mid\ !D \mid b[] \mid d[] \mid z[] \mid z'[]] \mid q'[\texttt{out}\ k.\texttt{in}\ q'_t]] \mid c_0[V(v_0-1) \mid ..] \mid c_1[V(v_1) \mid ..] \mid [\![\Delta]\!] \mid P_G$$

Finally, the ambient q' exits k by executing its capability $\texttt{out}\ k$ and the subprocess $k[n[!I \mid\ !D \mid b[\mathbf{0}] \mid d[\mathbf{0}] \mid z[\mathbf{0}] \mid z'[\mathbf{0}]]]$ is garbage-collected by the process P_G. The result indeed corresponds to the expected process $[\![(q', v_0-1, v_1)]\!]$.

We described above how the step relation of the two-counters machine can be simulated by some reductions of a process encoding some configuration. The sequences of reductions we described for each kind of transition relations are not the only possible ones for the process we considered. However, following some different sequences of reduction would either lead to a stuck process or would produce only processes that do not correspond to an encoding of the two-counters machine[1]. Our encoding is correct in the following sense:

Proposition 1. *For any two-counters machine $\mathcal{M} = (\mathcal{Q}, q_i, q_f, \Delta)$ and any arbitrary natural v, $\mathcal{M}$ accepts v iff $[\![(q_i, v, 0)]\!] \rightarrow^* [\![(q_f, 0, 0)]\!]$.*

Hence, as an immediate consequence using Theorem 1:

Theorem 2. *For any two arbitrary processes P, Q from the* `in/out` *ambient calculus, it is undecidable to test whether $P \rightarrow^* Q$.*

We believe that our encoding can easily be adapted to safe mobile ambients [14] from which name restriction, communication, the capability `open` and the co-capability $\overline{\texttt{open}}$ have been removed.

We claim that one of the sources of undecidability of the reachability problem is the rule (Str Repl Copy) from the structural congruence. On one hand, this rule can be used to exhibit a new process P from $!P$; this creates new possible interactions through reduction for this occurrence of P. On the other hand, it can be used to transform $!P \mid P$ into $!P$; we used this feature in our encoding to provide a garbage-collecting mechanism. Supporting our claim, we will see in the next section that if we drop this second possibility, then the reachability problem becomes decidable.

4 The reachability problem for a weaker reduction relation

In this section, we study the reachability problem for the `in/out` ambient calculus equipped with a weaker reduction relation. We show that for this new reduction relation, the reachability problem becomes decidable.

4.1 Definitions

The weaker reduction relation we consider here has been introduced in [1] [2]. Its main feature is to turn the axiom defining replication, that is $!P \equiv !P \mid P$ (Str Repl Copy), into an (oriented) reduction rule $!P \rightarrow P \mid !P$ (wRed Repl).

We consider the *weak structural congruence* $\cong$ defined as the least congruence relation over processes satisfying all the axioms defining $\equiv$ except (Str Repl Copy). We called this structural congruence weak as obviously $P \cong Q$ implies $P \equiv Q$ whereas the converse does not hold. Based on this weak structural congruence, we define a *weak reduction relation* as the least relation satisfying the following axioms:

[1] We prove this fact by defining a general shape matching all reachable processes and by applying an exhaustive inspection of all possibles reductions.

[2] In [19], the iteration is also defined by means of a reduction rule, but for explicit recursion instead of replication.

Weak Reduction: $P \rightarrow_{\mathrm{w}} Q$

$n[\mathtt{in}\ m.P \mid Q] \mid m[R] \rightarrow_{\mathrm{w}} m[n[P \mid Q] \mid R]$	(wRed In)
$m[n[\mathtt{out}\ m.P \mid Q] \mid R] \rightarrow_{\mathrm{w}} n[P \mid Q] \mid m[R]$	(wRed Out)
$!P \rightarrow_{\mathrm{w}} P \mid !P$	(wRed Repl)
$P \rightarrow_{\mathrm{w}} Q \Rightarrow P \mid R \rightarrow_{\mathrm{w}} Q \mid R$	(wRed Par)
$P \rightarrow_{\mathrm{w}} Q \Rightarrow n[P] \rightarrow_{\mathrm{w}} n[Q]$	(wRed Amb)
$P' \cong P, P \rightarrow_{\mathrm{w}} Q, Q \cong Q' \Rightarrow P' \rightarrow_{\mathrm{w}} Q'$	(wRed $\cong$)

This new reduction relation is strictly weaker than the one presented in Section 2:

Proposition 2. *For all processes P, Q if $P \rightarrow_w^* Q$ then $P \rightarrow^* Q$. Moreover, there exist two processes P' and Q' such that $P' \rightarrow^* Q'$ and $P' \not\rightarrow_w^* Q'$.*

Let us point out that if we consider additionally `open` capabilities and enrich the definition of $\rightarrow_{\mathrm{w}}$ with the rule $\mathtt{open}\ n.P \mid n[Q] \rightarrow_{\mathrm{w}} P \mid Q$ (Red Open) then the reachability problem for this weak reduction relation is undecidable: the encoding of the Post Correspondence Problem given in [7] provides a proof for this statement.

4.2 The reachability problem

We will show that the reachability problem is decidable for the weak reduction relation.

Theorem 3. *For all processes S and T, it is decidable to test whether $S \rightarrow_w^* T$.*

The rest of this section is devoted to the proof of Theorem 3. The main guidelines of this proof are as follows: first, we introduce a notion of normal form for processes for which we specialize the weak reduction relation; secondly, we show that the reachability problem for two arbitrary processes can be expressed as the reachability problem on their respective normal forms. Finally, we show how to reduce reachability problem for normal forms into markings reachability in Petri nets, a problem known to be decidable.

• **From weak reduction to weak reduction over normal forms:** As done in [12, 8], we consider the axioms $P \mid \mathbf{0} \cong P$, $!\mathbf{0} \cong \mathbf{0}$, $!!P \cong !P$ and $!(P \mid Q) \cong !P \mid !Q$ from the definition of $\cong$. We turn those axioms into a rewrite system $\mathcal{W}$ by orienting them from left to right and we denote $\rightsquigarrow_{\mathcal{W}}$ the AC-rewrite relation induced by $\mathcal{W}$ (taking into account associativity and commutativity for the parallel operator $|$). It can be shown that the AC-rewrite relation $\rightsquigarrow_{\mathcal{W}}$ is terminating and confluent. Hence, for any process P, there exists a unique (modulo associativity and commutativity for $|$) normal form $\tilde{P}$ of P wrt $\rightsquigarrow_{\mathcal{W}}$. This implies also that $\cong$ is decidable.

In the sequel we denote $=_{\mathsf{AC}}$ the equality relation over processes modulo associativity and commutativity for the parallel operator $|$.

We introduce a new reduction relation for normal forms. In particular, we require that any process in normal form is reduced to some normalized process. This reduction relation is denoted $\twoheadrightarrow$ and is given in the table below:

Normal Weak Reduction: $P \twoheadrightarrow Q$

$n[\mathtt{in}\ m.P] \mid m[\mathbf{0}] \twoheadrightarrow m[n[P]]$		(wRed In 1)
$n[\mathtt{in}\ m.P] \mid m[R] \twoheadrightarrow m[n[P] \mid R]$	if $R \neq \mathbf{0}$	(wRed In 2)
$n[\mathtt{in}\ m.\mathbf{0} \mid Q] \mid m[\mathbf{0}] \twoheadrightarrow m[n[Q]]$		(wRed In 3)

$n[\texttt{in}\ m.\mathbf{0} \mid Q] \mid m[R] \twoheadrightarrow m[n[Q] \mid R]$ if $R \neq \mathbf{0}$	(wRed In 4)
$n[\texttt{in}\ m.P \mid Q] \mid m[\mathbf{0}] \twoheadrightarrow m[n[P \mid Q]]$ if $P \neq \mathbf{0}$	(wRed In 5)
$n[\texttt{in}\ m.P \mid Q] \mid m[R] \twoheadrightarrow m[n[P \mid Q] \mid R]$ if $P \neq \mathbf{0}$ and $R \neq \mathbf{0}$	(wRed In 6)
$m[n[\texttt{out}\ m.P]] \twoheadrightarrow n[P] \mid m[\mathbf{0}]$	(wRed Out 1)
$m[n[\texttt{out}\ m.P] \mid R] \twoheadrightarrow n[P] \mid m[R]$	(wRed Out 2)
$m[n[\texttt{out}\ m.\mathbf{0} \mid Q]] \twoheadrightarrow n[Q] \mid m[\mathbf{0}]$	(wRed Out 3)
$m[n[\texttt{out}\ m.\mathbf{0} \mid Q] \mid R] \twoheadrightarrow n[Q] \mid m[R]$	(wRed Out 4)
$m[n[\texttt{out}\ m.P \mid Q]] \twoheadrightarrow n[P \mid Q] \mid m[\mathbf{0}]$ if $P \neq \mathbf{0}$	(wRed Out 5)
$m[n[\texttt{out}\ m.P \mid Q] \mid R] \twoheadrightarrow n[P \mid Q] \mid m[R]$ if $P \neq \mathbf{0}$	(wRed Out 6)
$!P \twoheadrightarrow P \mid !P$	(wRed Repl 1)
$!P \twoheadrightarrow !P \mid !P$	(wRed Repl 2)
$P \twoheadrightarrow Q \Rightarrow P \mid R \twoheadrightarrow Q \mid R$	(wRed Par)
$P \twoheadrightarrow Q \Rightarrow n[P] \twoheadrightarrow n[Q]$	(wRed Amb)
$P' =_{\mathsf{AC}} P, P \twoheadrightarrow Q, Q =_{\mathsf{AC}} Q' \Rightarrow P' \twoheadrightarrow Q'$	(wRed $=_{\mathsf{AC}}$)

Due to required normalization in presence of $\mathbf{0}$, several rules have been introduced for reductions of the `in` and `out` capabilities; moreover, one rule has been added for the reduction of replication; it aims to simulate the weak reduction $!P \cong !!P \rightarrow_{\mathsf{w}} !!P \mid !P \cong !P \mid !P$. It is easy to see that if P is in normal form and $P \twoheadrightarrow Q$ then Q is in normal form as well. Moreover, we have the following property:

Proposition 3. *For all processes P, Q, let $\tilde{P}, \tilde{Q}$ be their respective normal forms. Then $P \rightarrow_w^* Q$ iff $\tilde{P} \twoheadrightarrow^* \tilde{Q}$.*

Proposition 3 states that the reachability problem for the weak reduction relation can be reduced to a similar problem but restricted to normalized processes. This implies in particular that the use of weak structural congruence has been replaced by the simpler [3] relation of equality modulo associativity and commutativity.

• **From normal processes to Petri nets:** We first show here how to build up a Petri net from a normalized process: roughly speaking, this Petri net aims to encode all the possible reductions over this process. We will show later how to use this Petri net to solve the reachability problem for processes.

Note that applying a reduction over a process either increases the number of ambients in the process or leaves it unchanged: more precisely, when the rule (wRed Repl 1) or (wRed Repl 2) is applied on some process R at some subprocess P containing an ambient then the rule lets the number of ambients increased in the resulting process; other kinds of reduction steps leave the number of ambients unchanged. As we want to decide for two given normalized processes P and Q whether $P \twoheadrightarrow^* Q$, the target process Q is fixed and the number of its ambients is known. Therefore, this can be used to provide an upper-bound on the maximal number of applications of the rules (wRed Repl 1) and (wRed Repl 2) when applied to some subprocess containing an ambient. A similar argument doesn't hold for capabilities as they can be consumed by the reduction rules for the `in` and `out` capabilities.

This remark leads us to split a process into several parts; intuitively, one of those parts will be a context containing ambients whereas the other ones will be subprocesses without ambients. An *ambient context* C is a process in which may occur some holes,

[3] For equality modulo associativity and commutativity, every congruence class is finite.

noted as $\Box$. Moreover, we require that in any subcontext $!C'$ of C, C' contains some ambient. Together with ambient contexts, we consider substitutions mapping occurrences of holes to ambient-free processes. Hence, the application of one of these substitutions to an ambient context yields a process.

We will need to refer to a precise occurrence of replication, ambient, capability or hole $\Box$ within an ambient context or a process. Therefore, we are going to introduce a labeling for those objects to be able to distinguish any two of them. We assume for that a countable set of labels. We say that a process P or an ambient context C is *well-labeled* if any label occurs at most once in P or C. For an ambient context C, we define $\mathit{Amb}(C)$ as the multiset of ambients in C.

A labeled transition system: for the reachability problem $S \twoheadrightarrow^* T$, we consider C_S a well-labeled ambient context as well as a mapping θ_S from the set of holes in C_S to labeled ambient-free processes of the form $!P$ such that $\theta_S(C_S)$ is well-labeled and $\theta_S(C_S) = S$ ignoring labels. We are going to describe as a labeled transition system $L_{S,T}$ all possible reductions for the context C_S: this includes reductions of replications and capabilities contained in C_S as well as capabilities and replications from processes associated with the holes of the context.

We consider here a labeled transition system $L_{S,T}$ whose states are AC-equivalent classes of ambient contexts (for simplicity, we often identify a state as one of the representants of its class). We also define a mapping $\theta_{L_{S,T}}$ extending θ_S. Initially, $L_{S,T}$ contains (the equivalence class of) C_S as a unique state and $\theta_{L_{S,T}} = \theta_S$. We iterate the following construction steps until $L_{S,T}$ is unchanged (we assume that any reduction through (wRed In i), (wRed Out i), (wRed Repl 1) and (wRed Repl 2) uses implicitly (wRed Amb), (wRed Par) and (wRed $=_{\mathsf{AC}}$)):

1. for any ambient context C from $L_{S,T}$, for any labeled capability $\mathtt{cap}^w n$ ($\mathtt{cap} \in \{\mathtt{in}, \mathtt{out}\}$) in C if this capability can be executed using one of the rules (wRed In i) or (wRed Out i) leading to some ambient context C', then the state C' and a transition from C to C' labeled by $\mathtt{cap}^w n$ are added to $L_{S,T}$.
2. for any ambient context C from $L_{S,T}$, for any labeled replication $!^w$ in C such that this replication can be reduced using the rule (wRed Repl 1) (resp. (wRed Repl 2)), we define the ambient context C' as follows: C' is identical to C except that the subcontext $!^w C_a$ in C is replaced by $!^w C_a \mid \gamma(C_a)$ (resp. $!^w C_a \mid \gamma(!^w C_a)$) in C'; the mapping γ relabels C_a (resp. $!^w C_a$) with fresh labels, that is labels occurring neither in some other state of the currently built transition system $L_{S,T}$ nor in the currently built $\theta_{L_{S,T}}$; moreover, we require that C' is well-labeled. If $\mathit{Amb}(C') \subseteq \mathit{Amb}(T)$ then the state C' and a transition from C to C' labeled by $!^w_{①}$ (resp. $!^w_{②}$) are added to $L_{S,T}$. Additionally, we define $\theta'_{L_{S,T}}$ as an extension of $\theta_{L_{S,T}}$ such that for all $\Box^{w'}$ in C_a, (i) $\theta'_{L_{S,T}}(\gamma(\Box^{w'}))$ and $\theta_{L_{S,T}}(\Box^{w'})$ are identical ignoring labels, (ii) labels in $\theta'_{L_{S,T}}(\gamma(\Box^{w'}))$ are fresh in the currently built transition system $L_{S,T}$ and in $\theta_{L_{S,T}}$ and (iii) $\theta'_{L_{S,T}}(C')$ is well-labeled. Finally, we set $\theta_{L_{S,T}}$ to $\theta'_{L_{S,T}}$.
3. for any ambient context C from $L_{S,T}$, for any labeled hole $\Box^w$ in C and for any capability $\mathtt{cap}^{w'} n$ (with $\mathtt{cap} \in \{\mathtt{in}, \mathtt{out}\}$) in the process $\theta_{L_{S,T}}(\Box^w)$, we consider the ambient context C_m identical to C except that $\Box^w$ in C has been replaced by $\Box^w \mid \mathtt{cap}^{w'} n.\mathbf{0}$ in C_m. If this capability $\mathtt{cap}^{w'} n$ can be executed in C_m using one

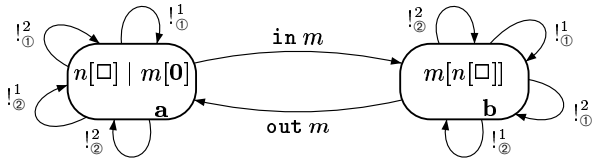

Fig. 1. A labeled transition system for the process $n[!^1\texttt{in}\ m.!^2\texttt{out}\ m.\mathbf{0}] \mid m[\mathbf{0}]$

of the rules (wRed In i) or (wRed Out i) leading to some ambient context C', then the state C' and a transition from C to C' labeled by $\texttt{cap}^{w'}\, n$ are added to $L_{S,T}$.

4. for any ambient context C from $L_{S,T}$, for any labeled hole $\Box^w$ in C associated by $\theta_{L_{S,T}}$ with a process of the form $!^{w'}P$, if the replication $!^{w'}$ can be reduced in the process $\theta_{L_{S,T}}(C)$ using the rule (wRed Repl 1), then for any replication $!^{w''}$ in $\theta_{L_{S,T}}(\Box^w)$, two transitions from C to itself, the first one labeled by $!^{w''}_{①}$ and the second one by $!^{w''}_{②}$ are added to $L_{S,T}$.

It should be noticed that in step 2 the reduction of a replication contained in the ambient context by means of the rule (wRed Repl 1) or (wRed Repl 2) is done only when the number of ambients in the resulting process is smaller than the number of ambients in the target process T. This requirement is crucial as it implies that the transition system $L_{S,T}$ has only finitely many states.

As an example, we give in Figure 1 the labeled transition system associated with the process $n[!^1\texttt{in}\ m.!^2\texttt{out}\ m.\mathbf{0}] \mid m[\mathbf{0}]$ (we omit in this process unnecessary labels).

One can also notice that the labeled transitions in $L_{S,T}$ for replications and capabilities from the ambient context correspond effectively to reductions performed on processes. Things are different for transitions corresponding to replications and capabilities contained in processes associated with holes, as shown in steps 3 and 4: these transitions are applied for any kind of those capabilities or replications independently of the fact that they are effectively at this point available to perform a transition.

We will solve this first by giving a model of processes corresponding to holes as Petri nets and then by synchronizing our two models: the Petri nets and the labeled transition system $L_{S,T}$.

From ambient-free processes to Petri nets: we show here how to build a Petri net from a labeled ambient-free process different from $\mathbf{0}$. For a set E, we denote $\mathcal{E}(E)$ the set of all multisets that can be built with elements from E. We recall that a Petri net is given by a 5-tuple $(\mathcal{P}, \mathcal{P}_i, \mathcal{T}, \mathit{Pre}, \mathit{Post})$ such that $\mathcal{P}$ is a finite set of *places*, $\mathcal{P}_i \subseteq \mathcal{P}$ is a set of *initial places*, $\mathcal{T}$ is a finite set of transitions and $\mathit{Pre}, \mathit{Post} : \mathcal{T} \to \mathcal{E}(\mathcal{P})$ are mappings from transitions to multisets of places. We say that an ambient-free process is *rooted* if it is of the form $\texttt{cap}^w\ n.P$ for $\texttt{cap} \in \{\texttt{in}, \texttt{out}\}$ or of the form $!^w P$. We define PN_P the Petri net associated with some rooted process P as follows: places for PN_P are precisely rooted subprocesses of P, and P itself is the unique initial place. Transitions are defined as the set of all capabilities $\texttt{in}^{\ w}\ n$, $\texttt{out}^{\ w'}\ n$ occurring in P and of all $!^w_{①}$,$!^w_{②}$ for replications $!^w$ occurring in P. Finally, *Pre* and *Post* are defined for all transitions as follows (for $\texttt{cap} \in \{\texttt{in}, \texttt{out}\}$):

- $\mathit{Pre}(\texttt{cap}^w\, n) = \{\texttt{cap}^w\, n.\mathbf{0}\}$ and $\mathit{Post}(\texttt{cap}^w\, n) = \varnothing$ if $\texttt{cap}^w\, n.\mathbf{0}$ is a place in PN_P.

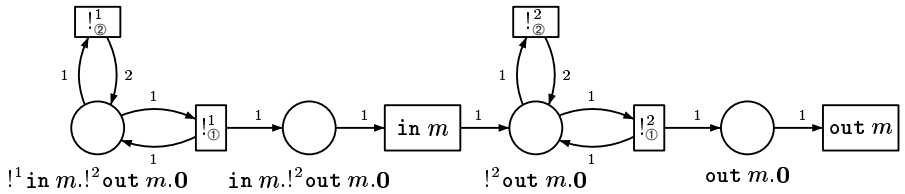

Fig. 2. A Petri net for the process $!^1\texttt{in}\ m.!^2\texttt{out}\ m.\mathbf{0}$

- *Pre*$(\texttt{cap}^w\ n) = \{\texttt{cap}^w\ n.(P_1 \mid \ldots \mid P_k)\}$ and *Post*$(\texttt{cap}^w\ n) = \{P_1, \ldots, P_k\}$ if $\texttt{cap}^w\ n.(P_1 \mid \ldots \mid P_k)$ is a place in PN_P ($P_1, \ldots, P_k$ being rooted processes).
- *Pre*$(!^w_①) =$ *Pre*$(!^w_②) = \{!^w P\}$, *Post*$(!^w_①) = \{!^w P, P\}$ and *Post*$(!^w_②) = \{!^w P, !^w P\}$ if $!^w P$ is a place in PN_P.

For $!^1\texttt{in}\ m.!^2\texttt{out}\ m.\mathbf{0}$, we obtain the Petri net given in Figure 2.

We will denote $PN_{\Box^w}$ the Petri net $PN(\theta_{L_{S,T}}(\Box^w))$, that is the Petri net corresponding to the rooted ambient-free process associated with $\Box^w$ by $\theta_{L_{S,T}}$.

We will show now how to combine the transition system $L_{S,T}$ and the Petri nets $PN_{\Box^w}$ into one single Petri net.

Combining the transition system and Petri nets: We first turn the labeled transition system $L_{S,T}$ into a Petri net $PN_L = (\mathcal{P}_L, \mathcal{P}^i_L, \mathcal{T}_L, \mathit{Pre}_L, \mathit{Post}_L)$. $\mathcal{P}_L$ is the set of states of $L_{S,T}$. $\mathcal{P}^i_L$ is a singleton set containing the state corresponding to C_S, the ambient context of S. The set of transitions $\mathcal{T}_L$ is the set of triples (s, l, s') where s, s' are states from $L_{S,T}$ with a transition labeled with l from s to s' in $L_{S,T}$. For all transitions $t = (s, l, s')$, *Pre*$(t) = \{s\}$ and *Post*$(t) = \{s'\}$.

We define the Petri net $PN_{S,T} = (\mathcal{P}_{S,T}, \mathcal{P}^i_{S,T}, \mathcal{T}_{S,T}, \mathit{Pre}_{S,T}, \mathit{Post}_{S,T})$ as follows: places (resp. initial places) from $PN_{S,T}$ are the union of places (resp. initial places) of PN_L and of each of the Petri nets $PN_{\Box^w}$ (for $\Box^w$ occurring in one of the states of $L_{S,T}$). Transitions of $PN_{S,T}$ are precisely transitions from PN_L. The mappings $\mathit{Pre}_{S,T}$ and $\mathit{Post}_{S,T}$ are defined as follows: for all transitions t (t being of the form (a, f, b)), (i) $\mathit{Pre}_{S,T}(t) = \{a\}$ and $\mathit{Post}_{S,T}(t) = \{b\}$ if f doesn't occur as a transition in any of the $PN_{\Box^w}$'s (for $\Box^w$ occurring in one of the states of $L_{S,T}$) and (ii) $\mathit{Pre}_{S,T}(t) = \{a\} \cup \mathit{Pre}_{\Box^w}(f)$ and $\mathit{Post}_{S,T}(t) = \{b\} \cup \mathit{Post}_{\Box^w}(f)$ if f is a transition of $PN_{\Box^w}$ ($\mathit{Pre}_{\Box^w}$ (resp. $\mathit{Post}_{\Box^w}$) being the mapping *Pre* (resp. *Post*) of $PN_{\Box^w}$).

We depict in Figure 3 the combination of the labeled transition system from Figure 1 and the Petri net from Figure 2.

Deciding reachability: We recall that for a Petri net $PN = (\mathcal{P}, \mathcal{P}_i, \mathcal{T}, \mathit{Pre}, \mathit{Post})$, a marking m is multiset from $\mathcal{E}(\mathcal{P})$. We say that a transition t is enabled by a marking m if *Pre*$(t) \subseteq m$. Firing the enabled transition t for the marking m gives the marking m' defined as $m' = (m \setminus \mathit{Pre}(t)) \cup \mathit{Post}(t)$ (where $\setminus$ stands for the multiset difference). A marking m' is reachable from m if there exists a sequence $m_0, \ldots, m_k$ of markings such that $m_0 = m$, $m_k = m'$ and for each m_i, m_{i+1}, there exists an enabled transition for m_i whose firing gives m_{i+1}.

Theorem 4. *[15] For all Petri nets P, for all markings m, m' for P, one can decide whether m' is reachable from m.*

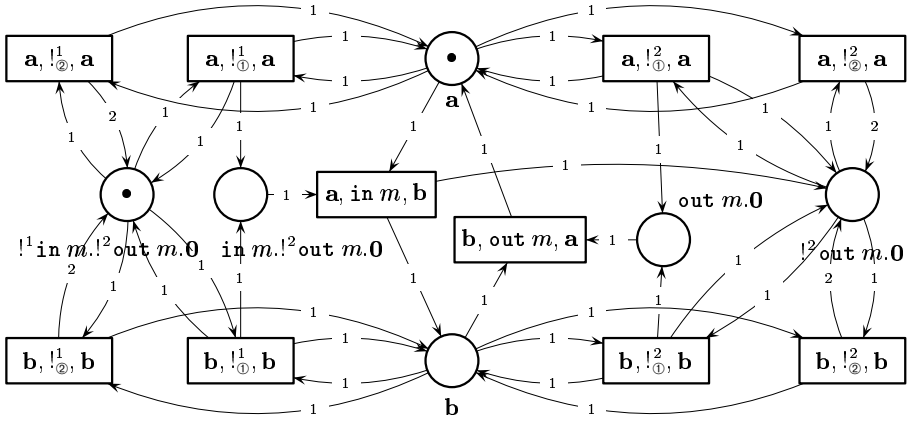

Fig. 3. The Petri net for the labeled process $n[!^1\texttt{in}\ m.!^2\texttt{out}\ m.\mathbf{0}] \mid m[\mathbf{0}]$

For the reachability problem $S \twoheadrightarrow^* T$, we consider the Petri net $PN_{S,T}$ and the initial marking m_S defined as $m_S = \mathcal{P}^i_{S,T}$. In Figure 3 is depicted the initial marking for the process $n[!^1\texttt{in}\ m.!^2\texttt{out}\ m.\mathbf{0}] \mid m[\mathbf{0}]$.

It should be noticed that for any marking m reachable from m_S, m contains exactly one occurrence of a place from $\mathcal{P}_L$. Roughly speaking, to any reachable marking corresponds exactly one ambient context. Moreover, the firing of one transition in the Petri net $PN_{S,T}$ simulates a reduction from $\twoheadrightarrow$. Thus, markings reachable from m_s correspond to normalized processes reachable from S.

We define now $\mathcal{M}_T$, the set of markings of $PN_{S,T}$ corresponding to T. Intuitively, a marking m belongs to $\mathcal{M}_T$ if m contains exactly one occurrence C of a place from $\mathcal{P}_L$ (that is, representing some ambient context) and in the context C, the holes can be replaced with ambient-free processes to obtain T. Moreover, each of those replication-free processes must correspond to a marking of the sub-Petri net associated with the hole it fills up. Formally, $\mathcal{M}_T$ is the set of markings m for $PN_{S,T}$ satisfying: (i) there exists exactly one ambient context C_m in m, (ii) ignoring labels, $\sigma_m(C_m)$ is equal to T modulo AC, for the substitution σ_m from holes $\square^w$ occurring in C_m to ambient-free processes defined as: $\sigma_m(\square^w) = P_1 \mid \ldots \mid P_k$ for $\{P_1, \ldots, P_k\}$ the multiset corresponding to the restriction of m to the places of $PN_{\square^w}$ and (iii) for all holes $\square^w$ occurring in some state of the transition system $L_{S,T}$ but not in C_m, the restriction of m to places of $PN_{\square^w}$ is precisely the set of initial places from $PN_{\square^w}$.

Proposition 4. *For the Petri net $PN_{S,T}$ built from a problem "$S \twoheadrightarrow^* T$", there are only finitely many markings corresponding to T, and their set $\mathcal{M}_T$ can be computed.*

The correctness of our reduction is stated in the following proposition which together with Theorem 4 implies Theorem 3:

Proposition 5. *For all normalized processes S, T, $S \twoheadrightarrow^* T$ iff there exists a marking m_T from $\mathcal{M}_T$ such that m_T is reachable from m_S in $PN_{S,T}$.*

5 On decision problems of name-convergence and model-checking

In this section, we investigate two problems closely related to reachability : the name-convergence problem and the model-checking problem.

5.1 The name-convergence problem

A process P *converges to a name* n if there exists a process P' such that P reduces to P' and P' is structurally congruent to $n[Q] \mid R$ (for some processes Q, R) [10]. The name-convergence problem is, given some process P and some name n, to decide whether P converges to the name n. We are going to show that this problem is not decidable for the two versions of the calculus we have considered so far.

In Section 3, we define the acceptance of an integer v by a two-counters machine $\mathcal{M}$ when $(q_i, v, 0) \vdash^*_{\mathcal{M}} (q_f, 0, 0)$ where q_i, q_f are respectively the initial and the final states of the machine $\mathcal{M}$ and $\vdash^*_{\mathcal{M}}$ is the reflexive-transitive closure of the step relation defined by the machine $\mathcal{M}$. This acceptance condition can be weakened as follows: we say that $\mathcal{M}$ accepts v if there exists two natural numbers v_1, v_2 such that $(q_i, v, 0) \vdash^*_{\mathcal{M}} (q_f, v_1, v_2)$. It is well-known that those two acceptance conditions lead to equally expressive two-counters machines [17].

Reconsidering the encoding given in Section 3, it can be proved that

Proposition 6. *For any two-counters machine $\mathcal{M} = (\mathcal{Q}, q_i, q_f, \Delta)$ and any natural v, the process $[\![(q_i, v, 0)]\!]$ converges to the name q_f iff there exist two natural numbers v_1, v_2, such that $[\![(q_i, v, 0)]\!] \rightarrow^* [\![(q_f, v_1, v_2)]\!]$.*

Now, for the weak calculus, it can be shown that

Proposition 7. *For all naturals v, v_1, v_2, if $[\![(q_i, v, 0)]\!] \rightarrow^* [\![(q_f, v_1, v_2)]\!]$ then there exists a process R such that $[\![(q_i, v, 0)]\!] \rightarrow^*_w [\![(q_f, v_1, v_2)]\!] \mid R$.*

Thus, using the fact that the acceptance of a natural number by an arbitrary two-counters machine is undecidable and Propositions 6 and 7, it holds that

Theorem 5. *The name-convergence problem is undecidable both for the* `in/out` *ambient calculus and for the weak* `in/out` *ambient calculus.*

5.2 The model-checking problem

The model-checking problem is to decide whether an ambient process satisfies (that is, is a model of) a given formula. Formulas that we consider here are the ones from the ambient logic [4]. The ambient logic is a modal logic used to specify properties of an ambient process; those modalities allow to speak both about time (that is, how a process can evolve by reduction) and space (that is, what is the shape of the tree description of a process). We will not describe the full logic, but focus on features of our interest.

Any process P satisfies the formula $\mathbf{T}$. A process P satisfies the formula $\Diamond\varphi$ if P can be reduced to some process Q ($P \rightarrow^* Q$ or $P \rightarrow^*_w Q$, depending on the considered calculus) such that Q satisfies the formula φ. A process P satisfies the formula $n[\varphi]$ if P is structurally congruent to some process $n[Q]$ ($P \equiv n[Q]$ or $P \cong n[Q]$, depending on the considered calculus) and Q satisfies φ. Finally, a process P satisfies the formula $\varphi \mid \psi$ if P is congruent to some process $Q \mid R$ and Q, R satisfy respectively φ and ψ.

Proposition 8. *A process P converges to the name n iff P satisfies $\Diamond(n[\mathbf{T}] \mid \mathbf{T})$.*

Using Theorem 5 and Proposition 8, we have

Theorem 6. *The model-checking problem for the* `in`/`out` *ambient calculus and for the weak* `in`/`out` *ambient calculus against the ambient logic is undecidable.*

References

1. T. Amtoft, A. J. Kfoury, and S. M. Pericás-Geertsen. What are polymorphically-typed ambients? In *10th European Symposium on Programming (ESOP 2001)*, LNCS 2028, pages 206–220. Springer, 2001.
2. M. Bugliesi, G. Castagna, and S. Crafa. Boxed ambients. In *Theoretical Aspects of Computer Software (TACS 2001)*, LNCS 2215. Springer, 2001.
3. N. Busi and G. Zavattaro. On the expressiveness of movement in pure mobile ambients. In *Foundations of Wide Area Network Computing*, ENTCS 66(3). Elsevier, 2002.
4. L. Cardelli and A.D. Gordon. Anytime, anywhere: Modal logics for mobile ambients. In *27th Symp. on Principles of Programming Languages (POPL'00)*, pages 365–377, 2000.
5. L. Cardelli and A.D. Gordon. Mobile ambients. *Theoretical Computer Science*, 240:177–213, 2000.
6. W. Charatonik, A. D. Gordon, and J.-M. Talbot. Finite-control mobile ambients. In *European Symposium on Programming (ESOP'02)*, LNCS 2305, pages 295–313. Springer, 2002.
7. W. Charatonik and J.-M. Talbot. The decidability of model checking mobile ambients. In *Computer Science Logic (CSL'01)*, LNCS 2142, pages 339–354. Springer, 2001.
8. S. Dal Zilio. Spatial congruence for ambients is decidable. In *6th Asian Computing Science Conference (ASIAN'00)*, volume 1961 of *LNCS*, pages 88–103. Springer, 2000.
9. J. Feret. Abstract interpretation-based static analysis of mobile ambients. In *Eighth International Static Analysis Symposium (SAS'01)*, LNCS 2126. Springer, 2001.
10. A. D. Gordon and L. Cardelli. Equational properties of mobile ambients. *Mathematical Structures in Computer Science*, 12:1–38, 2002.
11. R.R. Hansen, J.G. Jensen, F. Nielson, and H. Riis Nielson. Abstract interpretation of mobile ambients. In *Static Analysis (SAS'99)*, LNCS 1694, pages 134–148. Springer, 1999.
12. D. Hirschkoff. *Mise en œuvre de preuves de bisimulation*. PhD thesis, Ecole Nationale des Ponts et Chaussées, 1999.
13. D. Hirschkoff, E. Lozes, and D. Sangiorgi. Separability, expressiveness, and decidability in the ambient logic. In *Logic in Computer Science (LICS'02)*, pages 423–432. IEEE, 2002.
14. F. Levi and D. Sangiorgi. Controlling interference in ambients. In *27th Symp. on Principles of Programming Languages (POPL'00)*, pages 352–364, 2000.
15. E.W. Mayr. An Algorithm for the General Petri Net Reachability Problem. *SIAM Journal of Computing*, 13(3):441–460, 1984.
16. R. Milner, J. Parrow, and J. Walker. A calculus of mobile processes, I and II. *Information and Computation*, 100(1):1–40,41–77, 1992.
17. M. Minsky. Recursive Unsolvability of Post's Problem of "Tag" and others Topics in the Theory of Turing Machines. *Annals of Math.*, 74:437–455, 1961.
18. F. Nielson, H. Riis Nielson, R.R. Hansen, and J.G. Jensen. Validating firewalls in mobile ambients. In *Concurrency Theory (Concur'99)*, LNCS 1664, pages 463–477. Springer, 1999.
19. D. Teller, P. Zimmer, and D. Hirschkoff. Using ambients to control resources. In *CONCUR 2002—Concurrency Theory*, LNCS 2421, pages 288–303. Springer, 2002.
20. P. Zimmer. On the Expressiveness of Pure Safe Ambients. *Mathematical Structures of Computer Science*, 2002. To appear.