Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
ELSEVIER
Electronic Notes in Theoretical Computer Science 171 (2007) 69–79
www.elsevier.com/locate/entcs

On Reachability and Spatial Reachability in Fragments of BioAmbients

Giorgio Delzanno and Roberto Montagna

Dipartimento di Informatica e Scienze dell'Informazione
Università di Genova, Italy

Abstract

BioAmbients is a powerful model for representing various aspects of living cells. The model provides a rich set of operations for the movement and interaction of molecules. The richness of the language motivates the study of dialects of the full model and the comparison with other computational models. In this paper we investigate the limit between decidability and undecidability of two decision problems, namely reachability and spatial reachability, for semantic and syntactic fragments of BioAmbients providing movement capabilities and *merge*. Our results illustrate the power of *merge* with respect to the other movement operations of BA for properties like reachability. Furthermore, they establish an interesting connection between BioAmbients and other computational models like associative-commutative term rewriting and Petri nets with transfer arcs.

Keywords: Biological Systems, Term Rewriting, Reachability

1 Introduction

BioAmbients (BA for short) [9] is a model for biological systems inspired by the Mobile Ambients (MA) of Cardelli and Gordon [4] Ambients are used to build hierarchically structured biological processes. The BA language provides a rich set of capabilities for the movement of molecules between compartments and for modeling molecular interaction. Every capability comes with a corresponding co-capability. Furthermore, BA is equipped with a special operation for merging compartments, called *merge*. Given the richness of the language, it is important to the study the properties of dialects of the full model and to compare them with other computational models.

In this paper we focus our attention on the *reachability* and *spatial reachability* decision problems for pure public BA with a *weak reduction semantics* for replication (pBA_w). Pure public BA (pBA) is a fragment of BA with only movement capabilities and *merge*. Differently from the standard semantics, with the weak reduction of replication proposed in [1] the process $!P$ can only generate copies of P. The reachability problem consists in checking if a process P_0 can be reduced to a

1571-0661/$ – see front matter
doi:10.1016/j.entcs.2007.05.008

process P_1. The spatial reachability problem consists in checking if a process P_0 can be reduced to a process P_3 with the same ambient structure as P_1 and such that each ambient in P_3 has at least the same collection of local agents as the corresponding ambient in P_1. Spatial reachability has been introduced for Mobile Ambients in [2]. Our goal is to explore the limit between decidability and undecidability for these two decision problems by taking different assumptions on the syntax of pBA_w.

Concerning other computational models, we investigate here the connection between pBA and associative-commutative term rewriting. Specifically, we isolate a class of term rewriting, called TUC_M, in which terms represent unordered trees and rewriting rules have variables ranging over multisets of trees (multiset-variables). In this setting we define notions equivalent to reachability and spatial reachability and present an encoding from pBA to TUC_M that preserves their satisfiability. Furthermore, we show that spatial reachability is decidable in a particular fragment of TUC_M, called *structure preserving*, in which rewriting rules preserve the spatial structure of trees (i.e. when applied to a tree they cannot remove internal nodes). To obtain the decidability of reachability, we need an additional restriction on the merge-degree of a rewrite rule, i.e., on the number of occurrences of multiset-variables as siblings of internal nodes in a rewrite rule. Reachability turns out to be decidable for structure preserving rules with merge-degree equal to one and undecidable when the merge-degree is greater than one. The link between pBA and TUC_M can be used to transfer decidability results to BioAmbients.

Specifically, we first prove that reachability and spatial reachability are decidable in pBA_w without *merge* (pBA_w^{-m}). This fragment has the peculiarity that the number of ambients never decreases under applications of a reduction step. The proof exploits the encoding of pBA_w^{-m} into TUC_M. Reachability and spatial reachability become undecidable in pBA_w. The proof is based on an encoding of two counter machines in pBA_w. In the encoding we use nested ambients to simulate counters and the *merge* operation to implement the operations on counters. This negative result still holds for the *ambient preserving* fragment of PBA_w (pBA_w^a). In this fragment a syntactic restriction on the use of *merge* ensures that the number of ambients never decreases when applying a reduction step. Interestingly, this is the property needed to prove decidability of reachability in pBA_w^{-m}. The undecidability of pBA_w^a reachability follows from a weak simulation of two counter machines. The *merge* operation plays again a central role in the implementation of the operations on counters. Finally, we show that, perhaps surprisingly, spatial reachability is decidable in pBA_w^a. This results follows from an encoding of pBA_w^a spatial reachability into spatial reachability in structure preserving TUC_M. Consistently with the undecidability of reachability, for the encoding we need rules of merge-degree equal to two.

Related Work Reachability and Spatial Reachability have been studied for open-free fragments of Mobile Ambients with weak reduction and guarded replication in [1,2,3]. We are not aware of decidability results for the same properties in fragments of BioAmbients with merge. TUC_M is a generalization of the fragment of term rewriting we introduced in [5], called TUC, for studying reachability prob-

lems of Mobile Ambients. More precisely, TUC corresponds to the the subclass of TUC_M rules with merge-degree equal to one. The decidability of reachability for structure preserving with merge-degree equal to one has been proved in [5] by means of a reduction to Petri nets reachability. As a novel result with respect to [5], in the present paper we show that *spatial reachability* is decidable for structure preserving rules with any merge-degree and that reachability is undecidable for structure preserving $\text{TUC}_M$with merge-degree equal to two. Our decidability result is obtained via the encoding of TUC_M spatial reachability into coverability of Petri nets with *transfer arcs*. The latter problem has been proved to be decidable in [7].

Plan of the Paper In Section 2 we define pure public BA. In Section 3 we define TUC_M and the decidability result for spatial reachability. In Section 4 we define an encoding of reachability in pBA in TUC_M. In Section 5 we present decidability and undecidability results for fragments of pBA. In Section 6 we address some conclusions.

2 pBA: Pure Public BioAmbients

Processes in the pure (without communication) public (without name restriction) fragment of BA comply with the following grammar:

$$P ::= \mathbf{0} \quad | \quad [P] \quad | \quad M.P \quad | \quad P|P \quad | \quad !P$$
$$M ::= \mathit{enter}\ n \mid \mathit{accept}\ n \mid \mathit{exit}\ n \mid \mathit{expel}\ n \mid \mathit{merge}{+}\ n \mid \mathit{merge}{-}\ n$$

where n ranges over a denumerable set L of labels. $[P]$ denotes an *ambient*. The process $M.P$ denotes action prefixing, while $P|Q$ denotes the parallel composition of P and Q. The replication $!P$ denotes an arbitrary number of parallel copies of P. Finally, $\mathbf{0}$ denotes the null process. A *local agent* is a process of type $M.P$, $!P$ or $\mathbf{0}$. The operational semantics is defined by means of a structural congruence $\equiv$ and of a reduction relation $\hookrightarrow$. The structural congruence $\equiv$ is the smallest one satisfying

$$P \mid Q \equiv Q \mid P \qquad P \mid (Q \mid R) \equiv (P \mid Q) \mid R \qquad P \mid \mathbf{0} \equiv P \qquad !P \equiv !P \mid P$$

The reduction relation $\hookrightarrow$ is defined in Fig. 1 (notice that ! is not a context for reduction steps). We use $\hookrightarrow^*$ to denote the reflexive and transitive closure of the relation $\hookrightarrow$. Given processes P and Q, the *reachability problem* consists in deciding if $P \hookrightarrow^* Q$. In order to define spatial reachability we introduce the following ordering between processes.

For processes P and P', $P \preccurlyeq P'$ if there exist local agents P_i, Q_i for $i : 1, \ldots, n$, R_i for $i : 1, \ldots, r$, and processes A_i, B_i for $i : 1, \ldots, r$, $m, n, r \geq 0$, such that following conditions are all satisfied:

- $P \ = [\ A_1\ ] \mid \ \ldots\ \mid [\ A_r\ ] \mid P_1 \mid \ \ldots\ \mid P_n$
- $Q' = [\ B_1\ ] \mid \ \ldots\ \mid [\ B_r\ ] \mid Q_1 \mid \ \ldots\ \mid Q_n \mid R_1 \mid \ \ldots\ \mid R_m$
- $P_i \equiv Q_i$ for $i : 1, \ldots, n$ and $A_i \preccurlyeq B_i$ for $i : 1, \ldots, r$.

$$[merge+\ \ n.P \mid Q] \mid [merge- \ n.R \mid S] \hookrightarrow [P \mid Q \mid R \mid S]$$

$$[enter\ n.P \mid Q] \mid [accept\ n.R \mid S] \hookrightarrow [[P \mid Q] \mid R \mid S\ ]$$

$$[[exit\ n.P \mid Q] \mid expel\ n.R \mid S] \hookrightarrow [P \mid Q] \mid [R \mid S]$$

$$\frac{P \hookrightarrow Q}{P \mid R \hookrightarrow P \mid R} \qquad \frac{P \hookrightarrow Q}{n[P] \hookrightarrow n[Q]} \qquad \frac{P' \equiv P \ \ P \hookrightarrow Q \ \ Q \equiv Q'}{P' \hookrightarrow Q'}$$

Fig. 1. Reduction semantics for public MA.

For instance,

$$[merge+\ n.\mathbf{0}|\ [enter\ n.exit\ a.\mathbf{0}]\ ] \preccurlyeq$$
$$[merge+\ n.\mathbf{0}|merge-\ a.\mathbf{0}|\ [enter\ n.exit\ a.\mathbf{0} \mid exit\ a.\mathbf{0}]\ ].$$

Given P_1 and P_2, the spatial reachability problem consists in deciding if there exists P_3 such that $P_2 \preccurlyeq P_3$ and $P_1 \hookrightarrow^* P_3$.

Fragments of pBA We focus our attention on the following fragments.

pBA$_w$: The fragment with *weak reduction semantics* pBA$_w$ is obtained from pBA by transforming the congruence $!P \equiv !P|P$ into the oriented reduction rule (*copy*) defined as $!P \hookrightarrow !P|P$. In other words we forbid the absorb capability of replication.

pBA$_w^{-m}$: The *merge-free* fragment of pBA$_w^{-m}$ is obtained from pBA$_w$ by forbidding the use of $merge+$ and $merge-$.

pBA$_w^a$: The *ambient preserving* fragment pBA$_w^a$ is obtained from pBA$_w$ by restricting the syntax in the following way. Every occurrence of $merge+$ must have the following form: $merge+\ \ n.([Q] \mid R)$, for some label $n \in \mathcal{N}$ and some processes Q and R. This syntactic restriction ensures that the number of ambients never decreases when a reduction step is executed (the merging of two ambients is compensated by the creation of at least a new ambient).

3 TUC$_M$: A Fragment of AC Term Rewriting

In order to define the TUC$_M$ we need some preliminary definitions.

Ground Terms Given two *finite* sets of constants $\mathcal{N}$ and $\mathcal{Q}$ with $\mathcal{N} \cap \mathcal{Q} = \emptyset$, we use a constructor $n\langle \ldots \rangle$ to represent an ambient (internal node) with label $n \in \mathcal{N}$, an AC constructor $\mid$ to build multisets of trees (e.g. the sons of an internal node), ϵ to represent the empty multiset, and the finite set of constants in $\mathcal{Q}$ to represent processes (leaves). E.g., given $\mathcal{N} = \{n, m\}$ and $\mathcal{Q} = \{a, b\}$ the term $n\langle a \mid a \mid n\langle\epsilon\rangle \mid m\langle a \mid b\rangle\rangle$ can be viewed as an abstract representation of an ambient n with two subprocesses of type a and two subambients. Since ambients can be dynamically populated, we keep terms like $n\langle\epsilon\rangle$ (the empty ambient) distinguished from leaves in $\mathcal{Q}$.

Formally, the set TR of ground tree terms and the set MS of multisets of ground tree terms are defined as follows: $Q \subseteq TR$, $\epsilon \in MS$, if $t_1, \ldots, t_n \in TR$ then $t_1|\ldots|t_n \in MS$ for $n \geq 1$, if $m \in MS$ and $n \in \mathcal{N}$, then $n\langle m\rangle \in TR$.

Notice that, with a little bit of overloading, we use the same notation for a term t and the singleton multiset containing t. The multiset constructor $|$ is associative and commutative, i.e., $m_1|(m_2|m_3) = (m_1|m_2)|m_3$, and $m_1|m_2 = m_2|m_1$ for $m_1, m_2, m_3 \in MS$. Furthermore, $m \mid \epsilon = m$ for any $m \in MS$.

We use the special symbol *tuc* (not in $\mathcal{N}$) to represent a forest $t_1|\ldots|t_n$ as a single tree term $tuc\langle t_1|\ldots|t_n\rangle$. *tuc* never occurs in terms $t_1, \ldots, t_n$.

Restricted Terms with Multiset-Variables We consider here a restricted class of rewriting rules whose definition is based on two classes of terms called RT_L and RT_R. Given a denumerable set of multiset-variables $\mathcal{V} = \{X, Y, \ldots\}$,

- RT_L is the least set of terms of TR satisfying: $\mathcal{Q} \subseteq RT_L$; if $t_1, \ldots, t_n \in RT_L$, and $X \in \mathcal{V}$, then $n\langle t_1 \mid \ldots \mid t_n \mid X\rangle \in RT_L$ for $n, m \geq 0$.
- RT_R is the least set of terms satisfying: $\mathcal{Q} \subseteq RT_R$; if $t_1, \ldots, t_n \in RT_R$, and $X_1, \ldots, X_r \in \mathcal{V}$, then $n\langle t_1 \mid \ldots \mid t_n \mid X_1 \mid \ldots \mid X_r\rangle \in RT_R$ for $n, m, r \geq 0$.

We often use the abbreviated notation $n\langle t_1, \ldots, t_n|X_1, \ldots, X_m\rangle$ to denote the term $n\langle t_1|\ldots|t_n|X_1|\ldots|X_m\rangle$ where X_i is a variable for $i : 1, \ldots, m$ and t_i is a tree term for $i : 1, \ldots, n$.

Rewrite Rules A TUC_M rewrite rule $l \to r$ is such that

(i) $l = t_1 \mid \ldots \mid t_n$, and $t_i \in RT_L$ for $i : 1, \ldots, n$,

(ii) $r = t'_1 \mid \ldots \mid t'_m$ and $t'_i \in RT_R$ for $i : 1, \ldots, m$;

(iii) l and r have the same set V of variables;

(iv) each variable in V occurs once in l and once in r;

Notice that TUC_M forbids the use of rules like $R_1 = \ n\langle a\rangle \mid n\langle X\rangle \to n\langle a \mid X\rangle$, $R_2 = \ n\langle X \mid Y\rangle \to n\langle X\rangle \mid n\langle Y\rangle$, $R_3 = \ n\langle X\rangle \to n\langle X \mid X\rangle$.

A rule $l \to r$ is *structure preserving* if $IntNds(l) \leq IntNds(r)$, where $IntNds(t)$ denote the number of occurrences of labels in $\mathcal{N}$ in a term t.
Formally, $IntNds(t)$ is defined by induction on t as follows:

$$IntNds(\epsilon) = IntNds(X) = IntNds(q) = 0$$

for $X \in \mathcal{V}$ and $q \in \mathcal{Q}$, $IntNds(t_1|\ldots|t_k) = IntNds(t_1|\ldots|t_k|\,X) = \Sigma_{i=1}^{k} IntNds(t_i)$, and $IntNds(n\langle s\rangle) = IntNds(s) + 1$.

The *merge-degree* $\rightarrow$ of a rule $l \to r$ is defined as the largest number of multiset-variables occurring as sibling of internal nodes in r.
For instance the rule $n\langle a \mid a \mid X\rangle \mid m\langle a \mid b \mid Y\rangle \to n\langle m\langle X \mid Y\rangle\rangle$ is structure preserving with merge-degree equal to two. Notice that this rule is is not monotonic w.r.t. the size of terms (it removes some leaves).
The rule $n\langle X\rangle \mid m\langle a \mid b \mid Y\rangle \mid p\langle Z\rangle \to n\langle a \mid X \mid Y \mid Z\rangle$ is not structure preserving since it removes an internal node. Its merge-degree is three.

In the following we will call *structure preserving TUC_M with merge-degree k*, the fragment of TUC_M in which rules are structure preserving and have merge-degree less or equal than k.

Rewriting Relation We use the syntax $t[\,]$ to indicate a tree term with one occurrence of the constant $\circ$, and $t[s]$ to indicate the term obtained by replacing the constant $\circ$ in $t[\,]$ with s. Finally, we will use $var(t)$ to denote the set of variables in t. Given two ground TR terms $t_1 = tuc\langle m_1\rangle$ and $t_2 = tuc\langle m_2\rangle$ $t_1 \Rightarrow_{\mathcal{R}} t_2$ if and only if there exists a context $t[\,]$, two multisets of ground TR-terms m and m', a rule $l \to r$ in $\mathcal{R}$, and a mapping $\sigma : var(l) \to MS$ such that $t_1 \equiv t[\sigma(l)]$ and $t_2 \equiv t[\sigma(r)]$. We will use $\Rightarrow^*_{\mathcal{R}}$ to indicate the reflexive and transitive closure of the relation $\Rightarrow_{\mathcal{R}}$.

Given two ground terms $t_1 = tuc\langle m_1\rangle$ and $t_2 = tuc\langle m_2\rangle$, the *reachability problem* consists in deciding if $t_1 \Rightarrow^* t_2$.

In order to define spatial reachability we introduce the following ordering between trees. Given terms t and t', $t \sqsubseteq t'$ iff there exist terms $t_i, t'_i \notin \mathcal{Q}$ for $i : 1, \ldots, r$, and constants $q_i \in \mathcal{Q}$ for $i : 1, \ldots, n$ and $p_i \in \mathcal{Q}$ for $i : 1, \ldots, m$, $m, n, r \geq 0$ such that the following conditions are all satisfied:

- $t = n\langle q_1, \ldots, q_n, t_1, \ldots, t_r\rangle$,
- $t' = n\langle q_1, \ldots, q_n, p_1, \ldots, p_m, t'_1, \ldots, t'_r\rangle$,
- $t_i \sqsubseteq t'_i$ for $i : 1, \ldots, r$,

Given two ground terms $t_1 = tuc\langle m_1\rangle$ and $t_2 = tuc\langle m_2\rangle$, the *spatial reachability problem* consists in deciding if there exists a ground term t_3 such that $t_2 \sqsubseteq t_3$ and $t_1 \Rightarrow^*_{\mathcal{R}} t_3$.

In [5] we have proved that reachability and spatial reachability are decidable for TUC_M-theories with structure preserving rules of merge-degree equal to one (i.e. with no merging of multiset variables). The following property holds for rules with any merge degree.

Theorem 3.1 *Spatial reachability is decidable for TUC_M-theories with structure preserving rules of arbitrary merge-degree.*

Sketch of the proof. The proof is based on a reduction to the coverability problem for Petri nets with transfer arcs. For lack of space, we only give the intuition behind the construction. Given the initial term t_0 and the target term t_1, the construction of the Petri net is based on the following key ideas. The spatial structure of t_1 gives us an upper bound, namely $IntNds(t_1)$ on the number of internal nodes of terms occurring in a derivation $t_0 \Rightarrow^* t_2$ such that $t_1 \sqsubseteq t_2$. The Petri net has two types of places: places labeled by tree structures with at most $IntNds(t_1)$ internal nodes, and places labeled by leaves. Leaves are associated to internal nodes by means of special position labels. From every rewrite rule it is possible to extract a set of Petri net transitions that update the place encoding a tree structure, and rearrange the leaves according to the structure of the left- and right-hand side of the rule. Transfer arcs are used to encode rules with merge-degree greater than one. □

Furthermore, we have the following negative result.

Theorem 3.2 *Reachability is undecidable for TUC_M-theories with structure preserving rules and merge-degree equal to two.*

Proof. We exhibit an encoding of reachability for two counter machines (2CM). The instruction set of a 2CM with control states $s_1, \ldots, s_n$ and counters c_1 and c_2 consists of the intructions $INC_i(k,l)$ and $DEC_i(k,l,m)$ with the following semantics. When executed in state s_k, $INC_i(k,l)$ increments counter c_i and then move to state s_k, while $DEC_i(k,l,m)$ decrements c_i and then move to s_l if $c_i > 0$, and move to state s_m if $c_i = 0$. For simplicity, we consider a non-deterministic version of 2CM with separate operations for the test for zero and test for non-zero of a counter, and for the increment and decrement operations (the if-then-else instruction for decrement is non-deterministically simulated by two instructions defined on the same control location which uses the two tests). A counter c_i with value n is encoded as a term $c_i\langle t\rangle$ where t is a multiset with n occurrences of the leaf q. We encode a two counter machine M by using the following mapping from instructions to rules. The increment operation $INC_i(k,l)$ is encoded by the rule $s_k|c_i\langle X\rangle \to s_l|c_i\langle q|X\rangle$, the decrement operation for $c_i > 0$ is encoded by the rule $s_k|c_i\langle q|X\rangle \to s_l|c_i\langle X\rangle$, and for $c_i = 0$ by the rule $s_k|c_i\langle X\rangle|g\langle Y\rangle \to s_m|c_i\langle\epsilon\rangle|g\langle X|Y\rangle$. The term $g\langle\ldots\rangle$ is used to collect the content of a counter each time the test for zero is executed. If the test is executed when the counter is zero nothing is moved into the ambient g, otherwise we leave some garbage that we can use to distinguish bad simulations from good ones. Indeed, we have that the term $s_f|c_1\langle\epsilon\rangle|c_2\langle\epsilon\rangle|g\langle\epsilon\rangle$ is reachable from $s_0|c_1\langle\epsilon\rangle|c_2\langle\epsilon\rangle|g\langle\epsilon\rangle$ iff $\langle s_f, c_1 = 0, c_2 = 0\rangle$ is reachable from $\langle s_0, c_1 = 0, c_2 = 0\rangle$ in M.□

4 Encoding pBA_w (Spatial) Reachability in TUC_M

In this section we will show how to reduce the reachability problem for pBA_w to reachability in TUC_M. For this encoding, it is enough to use a very limited fragment of TUC_M. For instance, we will only consider trees with internal nodes all labelled by the same constant a. Before going into the details of the reduction, let us make some preliminary considerations on the semantics of BA. Let us first notice that we can work with a congruence relation applied only to context different from $!P$ (as for the reduction semantics). Let us now reformulate the axiom $P \mid \mathbf{0} \equiv P$ as the two reduction rules $P \hookrightarrow P \mid \mathbf{0}$ and $P \mid \mathbf{0} \hookrightarrow P$. Several computation steps of the modified semantics may correspond to one computation or congruence step in the original semantics. Reachability is preserved by the modified semantics: If Q is reachable from P_0 in the standard semantics, then there exists Q' reachable from P_0 in the modified semantics such that Q' is equivalent modulo the congruences for $\mathbf{0}$ to Q, and Q' is obtained by replacing every occurrences of a process $!R$ in Q with an equivalent process $!R'$ occurring in P_0.

Given a process term P, let us now define the set of replicated or sequential processes $Sub(P)$ (modulo associativity and commutativity of parallel) that may become active during a computation.
Formally, $Sub(\mathbf{0}) = \{\mathbf{0}\}$, $Sub([P]) = Sub(P)$, $Sub(!P) = \{!P\} \cup Sub(P)$, $Sub(P \mid Q) = Sub(P) \cup Sub(Q)$, $Sub(M.P) = \{M.P\} \cup Sub(P)$.

$$(merge)\ a\langle q_{merge+\ n.Q} \mid X\rangle \mid a\langle q_{merge-\ n.R} \mid Y\rangle \ \rightarrow\ a\langle T(Q) \mid T(R) \mid X \mid Y\rangle$$

$$(enter)\ a\langle q_{enter\ n.Q} \mid Y\rangle \mid a\langle q_{accept\ n.R} \mid Z\rangle \ \rightarrow\ a\langle a\langle T(Q) \mid Y\rangle \mid T(R) \mid Z\rangle$$

$$(accept)\ a\langle a\langle q_{exit\ n.Q} \mid Y\rangle \mid q_{expel\ n.R} \mid Z\rangle \ \rightarrow\ a\langle T(Q) \mid Y\rangle \mid a\langle q_{\mathbf{0}} \mid T(R) \mid Z\rangle$$

$$(copyt)\ \ q_{!Q} \rightarrow q_{!Q} \mid T(Q)$$

$$(zero)\ \ q \rightarrow q \mid q_{\mathbf{0}} \qquad a\langle X\rangle \rightarrow a\langle X\rangle \mid q_{\mathbf{0}} \qquad q \mid q_{\mathbf{0}} \rightarrow q \qquad a\langle X\rangle \mid q_{\mathbf{0}} \rightarrow a\langle X\rangle$$

Fig. 2. TUC_M-rules encoding pBA_w for $q_{M.Q}, q_{!Q} \in \mathcal{Q}$.

It is easy to check that $Sub(P)$ is a finite set. Furthermore, if $P \hookrightarrow^* Q$ using the modified reduction semantics, then $Sub(Q) \subseteq Sub(P) \cup \{\mathbf{0}\}$.

Let us now consider the reachability problem $P_0 \hookrightarrow^* P_1$. To encode this problem in TUC_M, we use terms in which leaves range over the finite set of constants $\mathcal{Q} = \{q_R \mid R \in Sub(P_0)\} \cup \{q_{\mathbf{0}}\}$.

The encoding of BA is built in a natural way by a mapping local P agents to a leaf q_P and an ambients $[Q]$ to the tree term $a\langle T(Q)\rangle$ where a is a special label used to denote membranes, and $T(Q)$ inductively defines the encoding of Q in TUC_M. Formally, given a process Q derived from P_0, we define the ground term $T(Q)$ by induction on Q as follows: $T(Q) = q_Q$ if $Q \in \{\mathbf{0}, M.Q_1, !Q_1\}$, $T([Q_1]) = a\langle T(Q_1)\rangle$, and $T(Q_1|Q_2) = T(Q_1)|T(Q_2)$.

The following properties then hold.

Proposition 4.1 *$P_0 \hookrightarrow^* P_1$ if and only if $tuc\langle T(P_0)\rangle \Rightarrow^* tuc\langle T(P_1)\rangle$.*

Proposition 4.2 *There exists P_2 such that $P_1 \preccurlyeq P_2$ and $P_0 \hookrightarrow^* P_2$ iff $tuc\langle T(P_0)\rangle \Rightarrow^* tuc\langle T(P_2)\rangle$ and $tuc\langle T(P_1)\rangle \sqsubseteq tuc\langle T(P_2)\rangle$.*

5 Reachability and Spatial Reachability in pBA_w

In this section we study the decidability of (spatial) reachability for the fragments pBA_w^{-m}, pBA_w, and pBA_w^a of pBA. The first property is as follows.

Theorem 5.1 *Reachability nd spatial reachability are decidable in pBA_w^{-m}.*

Proof. We first notice that the TUC_M-theory that encodes a reachability problem for pBA_w^{-m} consists of a finite set of structure preserving rules with merge-degree equal to one (all rules but *merge* in Fig. 2). Thus, the result follows by applying Prop. 4.1, Prop. 4.2 and the decidability of reachability in this fragment of term rewriting proved in [5]. □

Theorem 5.2 *Reachability and spatial reachability are undecidable in pBA_w.*

Proof. We exhibit an encoding of two counter machines. Given the set of control location $Loc = \{L_1, \ldots, L_k\}$, the encoding of a 2CM with instruction $I_1, \ldots, I_n$ and initial configuration $C_0 = \langle L, c_1 = 0, c_2 = 0\rangle$ is defined as follows

$$P_0 = Prog \mid Loc \mid [\![c_1 = 0]\!] \mid [\![c_2 = 0]\!],$$

where $Prog = [!\llbracket I_1 \rrbracket | \ldots | !\llbracket I_n \rrbracket]$, and $Loc = \llbracket L \rrbracket = [merge{-}\ L.\mathbf{0}]$. The encoding is defined using the set of labels $\mathcal{L} = Loc \cup \{a, b, z_1, z_2, c_1, c_2\}$. To represent $c_i = 0$, we use the following ambient $\llbracket c_i = 0 \rrbracket ::= [!exit\ z_i.\mathbf{0} \mid !merge{-}\ z_i.\mathbf{0}]$ for $i : 1, 2$. To represent $c_i = k$ with $k > 0$, we use the ambient $\llbracket c_i = k \rrbracket ::= [merge{-}\, c_i.\mathbf{0} \mid \llbracket c_i = k - 1 \rrbracket]$ for $i : 1, 2$. The encoding of the instructions is defined as follows.

$I = DEC_i(L, M)$, $c_i = 1$: $\llbracket I \rrbracket = merge{+}\ L.(\mathbf{A_1} \mid expel\ a.\mathbf{0})$, where $\mathbf{A_1} = [exit\ a.merge{+}\ c_i.expel\ z_i.merge{-}\ M.\mathbf{0}]$. The Loc ambient is first merged with the $Prog$ ambient using the synchronization label L. This action creates the ambient $\mathbf{A_1}$ that is expelled by the merged ambients immediately after. $\mathbf{A_1}$ is merged with the ambient c_i. The resulting ambient expels the z_i ambient ($c_i = 1$) and then becomes a new ambient encoding the new location $\llbracket M \rrbracket$.

$I = DEC_i(L, M)$, $c_i > 1$: $\llbracket I \rrbracket = merge{+}\ L.(\mathbf{A_1} \mid expel\ a.\mathbf{0})$, where $\mathbf{A_1} = [exit\ a.merge{+}c_i.(\mathbf{A_2} \mid expel\ a.merge{-}M.\mathbf{0})]$, $\mathbf{A_2} = [merge{+}c_i.exit\ a.merge{-}c_i.\mathbf{0}]$. As in the previous case the Loc ambient is first merged with the $Prog$ ambient using the synchronization label L (the current location). This action creates the ambient $\mathbf{A_1}$ which is expelled immediately after. $\mathbf{A_1}$ is merged with the ambient c_i. A new ambient $\mathbf{A_2}$ is created inside the resulting merged ambient say $\mathbf{A_1} + c_i$. $\mathbf{A_2}$ is merged with the c_i ambient at the same level and the resulting ambient is moved at the top level (it represents $c_i - 1$) while $\mathbf{A_1} + c_i$ becomes the ambient $\llbracket M \rrbracket$.

$I = INC_i(L, M)$, $c_i = 0$: then $\llbracket I \rrbracket = merge{+}\ L.(\mathbf{A_1} \mid \mathbf{B_1} \mid expel\ a.expel\ a.\mathbf{0})$, where $\mathbf{A_1} = ([exit\ a.merge{+}\ z_i.enter\ a.\mathbf{A_2}] \mid expel\ b.\mathbf{0})$,
$\mathbf{B_1} = [exit\ a.accept\ a.expel\ a.merge{-}\ c_i.\mathbf{0}]$,
and $\mathbf{A_2} = [exit\ b.exit\ a.merge{-}\ M.\mathbf{0}]$. As for DEC the Loc ambient is first merged with the $Prog$ via L (the current location). This action creates the ambients $\mathbf{A_1}$ and $\mathbf{B_1}$ that are expelled immediately after. $\mathbf{A_1}$ is merged with the ambient z_i and then enters inside $\mathbf{B_1}$ where it releases an ambient $\mathbf{A_2}$. $\mathbf{A_2}$ is expelled by the two nested ambients and, thus, moved at the top level as the new location $\llbracket M \rrbracket$. In the meantime $\mathbf{B_1}$ creates a local agent $merge{-}\ c_i.\mathbf{0}$ to become $\llbracket c_i = 1 \rrbracket$.

$I = INC_i(L, M)$, $c_i > 1$: then $\llbracket I \rrbracket = merge{+}\ L.(\mathbf{A_1} \mid \mathbf{B_1} \mid expel\ a.expel\ a.\mathbf{0})$, where $\mathbf{A_1} = ([exit\ a.merge{+}\ c_i.enter\ a.(\mathbf{A_2} \mid merge{-}\ c_i.\mathbf{0})] \mid expel\ b.\mathbf{0})$, $\mathbf{A_2} = [exit\ b.exit\ a.merge{-}\ M.\mathbf{0}]$, $\mathbf{B_1} = [exit\ a.accept\ a.expel\ a.merge{-}\ c_i.\mathbf{0}]$. The tests $c_i = 0$ and $c_i > 0$ are simulated by using merge steps either with label z_i or with label c_i.

$I = TSTZ_i(L, M)$: then $\llbracket I \rrbracket = merge{+}\ L.(\mathbf{A_1} \mid expel\ a.\mathbf{0})$, where
$\mathbf{A_1} = [exit\ a.merge{+}\ z_i.(\mathbf{A_2} \mid expel\ a.\mathbf{0})]$, $\mathbf{A_2} = [exit\ a.merge{-}\ M.\mathbf{0}]$.

$I = TSTNZ_i(L, M)$: then $\llbracket I \rrbracket = merge{+}\ L.(\mathbf{A_1} \mid accept\ a.\mathbf{0})$, where $\mathbf{A_1} = [exit\ a.merge{+}\ c_i.(merge{-}\ c_i.\mathbf{0} \mid \mathbf{A_2} \mid accept\ a.\mathbf{0})]$, $\mathbf{A_2} = [exit\ a.merge{-}\ M.\mathbf{0}]$.

The 2CM reachability problem from C_0 to C_0 (a variation of the general reachability problem that it is still undecidable) can be reduced then, to the reachability problem $P_0 \hookrightarrow^* P_0$. Furthermore, since the only garbage introduced by the encoding is due to possible duplication of banged local agents, we have that that $P_0 \rightarrow^* P_1 \sqsupseteq P_0$ if and only if $P_0 \rightarrow^* P_0$. Since 2CM reachability is undecidable, we have that reachability and spatial reachability are both undecidable. □

The second negative results concerns reachability in the fragment pBA^a_w in which

merge is allowed only if it does not reduce the total number of ambients.

Theorem 5.3 *Reachability is undecidable in* pBA^a_w.

Proof. We exhibit a weak encoding of 2CMs. Let M be a 2CM with list of instructions $I_1, \ldots, I_n$. The current configuration is encoded using 5 ambients that we will label as *Prog*, *Loc*, C_1, C_2, and G: *Prog* contains the encoding of the instructions, *Loc* keeps track of the current control location, C_1, C_2 keep track of the current values of the counters, G has a subambient H needed to collect (and keep separated from the other ambients) all local agents representing "units" when the zero-test is weakly simulated. Specifically, the encoding of a 2CM with instruction $I_1, \ldots, I_n$ and initial configuration $C_0 = \langle L, c_1 = 0, c_2 = 0\rangle$ is defined as $P_0 = Prog \mid Loc \mid [\![c_1 = 0]\!] \mid [\![c_2 = 0]\!] \mid G$, where $Prog = [!\![I_1]\!] \ldots |![\![I_n]\!]]$, $Loc = [\![L]\!] = [merge- \;\; L.\mathbf{0}]$, $G = [!accept\ g.\mathbf{0} \mid H]$, and $H = [!merge- \; h.\mathbf{0}]$. To represent $c_i = k$ we define the ambient $[\![c_i = k]\!] = [merge- \;\; z_i.\mathbf{0} \mid P_k]$, where P_k is a parallel with k occurrences of the local agent $merge- \;\; c_i.\mathbf{0}$ for $i : 1, 2$.
The encoding of the instructions is defined as follows.

If $I = [\![DEC_i(L, M)]\!]$, then $[\![I]\!]$ is defined as $merge+ \; L.(\mathbf{A_1} \mid expel\ a.\mathbf{0})$, where $\mathbf{A_1} = [exit\ a.merge+ \;\; c_i.(\mathbf{A_2} \mid expel\ a.\mathbf{0})]$ and $\mathbf{A_2} = [exit\ a.merge- \;\; M.\mathbf{0}]$. The intuition of the previous definition is as follows. The *Loc* ambient is first merged with the *Prog* ambient using the synchronization label L (the current location). This action creates the ambient $\mathbf{A_1}$ that is expelled by the merged ambients immediately after. $\mathbf{A_1}$ is merged with the ambient c_i (thus consuming a "unit", i.e., a local agent $merge- \;\; c_i.\mathbf{0}$). The ambient $\mathbf{A_2}$ is created inside the resulting merged ambients and expelled to become $[\![M]\!]$.

If $I = INC_i(L, M)$, then $[\![I]\!]$ is defined as $merge+ \; L.(\mathbf{A_1} \mid expel\ a.\mathbf{0})$, where $\mathbf{A_1} = [exit\ a.merge+ \;\; c_i.(\mathbf{A_2} \mid expel\ a.\mathbf{0} \mid merge- \;\; c_i.\mathbf{0} \mid \;\; merge- \;\; c_i.\mathbf{0})]$, and $\mathbf{A_2} = [exit\ a.merge- \; M.\mathbf{0}]$. Again the *Loc* ambient is first merged with the *Prog* ambient via L. This action creates the ambient $\mathbf{A_1}$ that is expelled by the merged ambients immediately after. $\mathbf{A_1}$ is merged with the ambient c_i (thus consuming a "unit", i.e., a local agent $merge- \;\; c_i.\mathbf{0}$). The ambient $\mathbf{A_2}$ is created inside the resulting ambient, say $\mathbf{A_1} + c_i$, and expelled to become $[\![M]\!]$. In the meantime two new "units" are release inside $\mathbf{A_1} + c_i$ (one to compensate the unit consumed to execute the merge, and one for the increment).

The encoding of the zero test is more tricky, since it exploits the ambient we called G (garbage) at the begininning of the proof.
If $I = TSTZ_i(L, M)$, then $[\![I]\!] = merge+ \; L.(\mathbf{A_1} \mid expel\ a.\mathbf{0})$, where
$\mathbf{A_1} = [exit\ a.merge+ \;\; z_i.(\mathbf{A_2} \mid enter\ g.P \mid expel\ b.expel\ d.\mathbf{0})]$,
$P = merge+ \;\; h.(\mathbf{A_3} \mid expel\ a.expel\ c.\mathbf{0})$, $\mathbf{A_2} = [exit\ a.exit\ b.merge- \;\; z_i.\mathbf{0}]$, and $\mathbf{A_3} = [exit\ c.exit\ d.merge- \;\; M.\mathbf{0}]$. The intuition is as follows. The *Loc* ambient is first merged with the *Prog* ambient via L. This action creates the ambient $\mathbf{A_1}$ that is expelled by the merged ambients immediately after. $\mathbf{A_1}$ is merged with the ambient c_i via the label z_i (used only for the zero-test). $\mathbf{A_2}$ (that will become $[\![c_i = 0]\!]$) is released inside the resulting ambient, we will refer to as $\mathbf{A_1} + c_i$. At this stage, $\mathbf{A_1} + c_i$ enters G while creating another internal ambient $\mathbf{A_3}$ (that will

become $[\![M]\!]$), and the merges with H. As a last step, $\mathbf{A_2}$ and $\mathbf{A_3}$ are moved at the top level in sequence. If the counter c_i was not zero, then the local agents inside c_i remain blocked inside the subambient H of G. This way they cannot interact with the other ambients at the top level.
Finally, if $I = TSTNZ_i(L, M)$, then $[\![I]\!]$ is defined as $merge+\ L.(\mathbf{A_1} \mid accept\ a.\mathbf{0})$, where $\mathbf{A_1} = [exit\ a.merge+\ z_i.(\mathbf{A_2} \mid merge-\ c_i.\mathbf{0} \mid expel\ a.\mathbf{0})]$, and $\mathbf{A_2} = [exit\ a.merge-\ M.\mathbf{0}]$.
By means of the previous encoding, we can show that the 2CM reachability problem from C_0 to C_0 can be reduced then, to the reachability problem $P_0 \hookrightarrow^* P_0$. □

While reachability in pBA^a_w is undecidable, we can prove that spatial reachability remains decidable even in presence of the *merge* rule.

Theorem 5.4 *Spatial reachability is decidable for pBA^a_w.*

Proof. The TUC_M-rules that encode a reachability problem for pBA^a_w consists of a finite set of structure preserving rules with merge-degree two. Thus, the result follows by applying Prop. 4.2 and Theorem 3.1. □

6 Conclusions

In this paper we have investigated in the decidability/undecidability of reachability and spatial reachability for public fragments of BioAmbients with weak reduction for replication. Our results illustrate the power of the *merge* operation. Its presence can turn a minimal fragment of public BioAmbients into a Turing equivalent model. Furthermore, they establish an interesting connection between BioAmbients and other computational models like associative and commutative term rewriting and Petri nets with transfer arcs. This connection can be used to define executable specifications of biological systems by means of tools like Elan and Maude (see e.g. [10]). We plan to investigate this direction in our future research.

References

[1] I. Boneva and J.-M. Talbot. When ambients cannot be opened! TCS 333(1-2): 127-169, 2005.

[2] N. Busi and G. Zavattaro. Deciding reachability in Mobile Ambients. ESOP '05: 248-262.

[3] N. Busi and G. Zavattaro. Deciding Reachability in Boxed Ambients. ICTCS '05: 143-159.

[4] L. Cardelli and A. D. Gordon. Mobile ambients. TCS 240(1): 177-213, 2000.

[5] G. Delzanno and R. Montagna. Reachability analysis of Mobile Ambients in fragments of AC term rewriting. Technical Report DISI-05-16, Università di Genova, 2006.

[6] M. Dauchet, S. Tison. The Theory of ground rewrite systems is decidable. LICS 1990: 242-248.

[7] J. Esparza, A. Finkel, and R. Mayr. On the verification of broadcast protocols. LICS 1999: 352-359.

[8] R. Mayr and M. Rusinowitch. Reachability is decidable for ground AC rewrite systems. INFINITY'98.

[9] A. Regev, E. M. Panina, W. Silverman, L. Cardelli, E. Y. Shapiro. BioAmbients: an abstraction for biological compartments. TCS 325(1): 141-167, 2004.

[10] F. Rosa Velardo, C. Segura, A. Verdejo. Typed Mobile Ambients in Maude. ENTCS 147(1): 135-161, 2006.