Reachability of Hennessy-Milner Properties for Weakly Extended PRS

Mojmír Křetínský, Vojtěch Řehák, and Jan Strejček

Faculty of Informatics, Masaryk University, Brno, Czech Republic
{kretinsky,rehak,strejcek}@fi.muni.cz

Abstract. We examine the problem whether a given weakly extended process rewrite system (wPRS) contains a reachable state satisfying a given formula of Hennessy–Milner logic. We show that this problem is decidable. As a corollary we observe that the problem of strong bisimilarity between wPRS and finite-state systems is decidable. Decidability of the same problem for wPRS subclasses, namely PAN and PRS, has been formulated as an open question, see e.g. [Srb02]. We also strengthen some related undecidability results on some PRS subclasses.

1 Introduction

Current software systems often exhibit an evolving structure and/or operate on unbounded data types. Hence automatic verification of such systems usually requires to model them as infinite-state ones. Various modeling formalisms suited to different kinds of applications have been developed with their respective advantages and limitations.

Here we employ the classes of infinite-state systems defined by term rewrite systems and called *Process Rewrite Systems* (PRS) as proposed by Mayr [May00]. PRS subsume a variety of the formalisms studied in the context of formal verification. *Petri nets* (PN), *pushdown processes* (PDA), and process algebras like BPA, BPP, or PA all serve to exemplify this. The relative expressive power of various process classes has been studied, especially with respect to strong bisimulation; see [BCS96, Mol96] and [May00] showing the strictness of the PRS hierarchy. Their relevance for modeling and analysing programs is shown e.g. in [Esp02], for automatic verification see e.g. surveys [BCMS01, KJ02, Srb02].

Expressiveness of (most of) the PRS classes can be increased by adding a finite-state control unit to the PRS rewriting mechanism, which results in so-called state-extended PRS (sePRS) classes, see e.g. [JKM01, HM01]. We have extended the PRS hierarchy by the sePRS classes and refined this extended hierarchy by introducing PRS equipped with a weak finite-state unit (wPRS, inspired by weak automata [MSS92]) [KŘS04b, KŘS04a].

Research on the expressive power of process classes has been accompanied by exploring algorithmic boundaries of various verification problems. In this paper we mainly focus on model checking some (fragments of) simple branching time logics, namely EF and EG, on the process classes mentioned above.

R. Ramanujam and S. Sen (Eds.): FSTTCS 2005, LNCS 3821, pp. 213–224, 2005.

First, we note that the *reachability problem*, i.e. to decide whether a given state is reachable from the initial one, is decidable for the classes of PRS [May00] and wPRS [KŘS04a], while it is undecidable for sePA [BEH95]. All the problems mentioned below remain undecidable on the sePA class due to its Turing power.

A *reachability property problem*, for a given system Δ and a given formula φ, is to decide whether $\mathsf{EF}\varphi$ holds in the initial state of Δ. Hence, these problems are parametrized by the class to which the system Δ belongs, and by the type of the formula φ. In most of practical situations, φ specifies error states and the reachability property problem is a formalization of a natural verification problem whether some error state is reachable in a given system.

We recall that the (full) EF logic is decidable for PAD [May98] (PAD subsumes both PA and PDA). It is undecidable for PN [Esp94]; an inspection of the proof moves this undecidability border down to seBPP (also known as *multiset automata*, MSA). If we consider the *reachability HM property problem*, i.e. the reachability property problem where φ is a formula of Hennessy–Milner logic (HM formula), then this problem has been shown to be decidable for the classes of PN [JM95] and PAD [JKM01]. We lift the decidability border for this problem to the wPRS class. This results also moves the decidability border for the *reachability simple property problem*, i.e. the reachability property problem where φ is a HM formula without any nesting of modal operators $\langle a \rangle$, as the problem has been know to be decidable for PRS [May00] so far.

Let us recall that the (full) EG logic is decidable for PDA (a consequence of [MS85] and [Cau92]), whilst undecidability has been obtained for its $\mathsf{EG}\varphi$ fragment on (deterministic) BPP [EK95], where φ is a HM formula. We show that this problem remains undecidable on (deterministic) BPP even if we restrict φ to a HM formula without nesting of modal operators $\langle a \rangle$.

As a corollary of our main result, i.e. decidability of the reachability HM property problem for wPRS, we observe that the problem of strong bisimilarity between wPRS systems and finite-state ones is decidable. As PRS and its subclasses are proper subclasses of wPRS, it follows that we positively answer the question of the reachability HM property problem for the PRS class and hence the questions of bisimilarity checking the PAN and PRS processes with finite-state ones, which have been open problems, see for example [Srb02]. Their relevance to program specification and verification is advocated, for example, in [JKM01, KS04].

The outline of the paper is as follows. In the next section we recall some basic notions including syntax and semantics of (extended) PRS. In Section 3 we show that the problem of reachability HM property is decidable for wPRS. Some consequences and further results are discussed in Section 4.

2 Preliminaries

2.1 PRS and Its Extensions

Let $Const = \{X, \ldots\}$ be a set of *process constants*. The set of *process terms* (ranged over by $t, \ldots$) is defined by the abstract syntax $t ::= \varepsilon \mid X \mid t.t \mid t \| t$,

where ε is the *empty term*, $X \in Const$ is a process constant; and '.' and '$\|$' mean *sequential* and *parallel compositions* respectively. We always work with equivalence classes of terms modulo commutativity and associativity of '$\|$', associativity of '.', and neutrality of ε, i.e. $\varepsilon.t = t.\varepsilon = t\|\varepsilon = t$. We distinguish four *classes of process terms* as:

1 – terms consisting of a single process constant only, in particular $\varepsilon \notin 1$,
S – *sequential* terms - terms without parallel composition, e.g. $X.Y.Z$,
P – *parallel* terms - terms without sequential composition, e.g. $X\|Y\|Z$,
G – *general* terms - terms without any restrictions, e.g. $(X.(Y\|Z))\|W$.

Let $M = \{o, p, q, \ldots\}$ be a set of *control states* and $Act = \{a, b, c, \ldots\}$ be a set of *actions*. Let $\alpha, \beta \in \{1, S, P, G\}, \alpha \subseteq \beta$ be the classes of process terms. An *(α, β)-sePRS (state extended process rewrite system)* Δ is a tuple (R, p_0, t_0), where

- R is a finite set of *rewrite rules* of the form $(p, t_1) \overset{a}{\hookrightarrow} (q, t_2)$, where $t_1 \in \alpha$, $t_1 \neq \varepsilon$, $t_2 \in \beta$, $p, q \in M$, and $a \in Act$,
- a pair $(p_0, t_0) \in M \times \beta$ forms the distinguished *initial state* of the system.

Sets of control states and process constants occurring in rewrite rules or in the initial state of Δ are denoted by $M(\Delta)$ and $Const(\Delta)$ respectively.

An (α, β)-sePRS $\Delta = (R, p_0, t_0)$ represents a labelled transition system the states of which are pairs (p, t) such that $p \in M(\Delta)$ is a control state and $t \in \beta$ is a process term over $Const(\Delta)$. The transition relation $\longrightarrow$ is the least relation satisfying the following inference rules:

$$\frac{((p, t_1) \overset{a}{\hookrightarrow} (q, t_2)) \in \Delta}{(p, t_1) \overset{a}{\longrightarrow} (q, t_2)} \qquad \frac{(p, t_1) \overset{a}{\longrightarrow} (q, t_2)}{(p, t_1\|t_1') \overset{a}{\longrightarrow} (q, t_2\|t_1')} \qquad \frac{(p, t_1) \overset{a}{\longrightarrow} (q, t_2)}{(p, t_1.t_1') \overset{a}{\longrightarrow} (q, t_2.t_1')}$$

Sometimes we use $\longrightarrow_\Delta$ or $\longrightarrow_R$ to emphasize that we mean the transition relation corresponding to Δ or the relation generated by a set of rules R, respectively. To shorten our notation we write pt in lieu of (p, t). The transition relation can be extended to finite words over Act in a standard way. A state qt_2 is *reachable from a state* pt_1, written $pt_1 \overset{*}{\longrightarrow} qt_2$, if there is $\sigma \in Act^*$ such that $pt_1 \overset{\sigma}{\longrightarrow} qt_2$. We say that a state is *reachable* if it is reachable from the initial state.

An (α, β)-sePRS where $M(\Delta)$ is a singleton is called *(α, β)-PRS (process rewrite system)*. In such systems we omit the single control state from rules and states. An (α, β)-sePRS Δ is called a *process rewrite system with weak finite-state control unit* or a *weakly extended process rewrite system*, written (α, β)-wPRS, if there exists a partial order $\leq$ on $M(\Delta)$ such that each rule $pt_1 \overset{a}{\hookrightarrow} qt_2$ of Δ satisfies $p \leq q$.

Some classes of (α, β)-PRS correspond to widely known models as *finite-state systems* (FS, $(1, 1)$-PRS), *basic process algebras* (BPA, $(1, S)$-PRS), *basic parallel processes* (BPP, $(1, P)$-PRS), *process algebras* (PA, $(1, G)$-PRS), *pushdown processes* (PDA, (S, S)-PRS, see [Cau92] for justification), and *Petri nets* (PN, (P, P)-PRS). The classes (S, G)-PRS, (P, G)-PRS and (G, G)-PRS were

introduced and named as PAD, PAN, and PRS by Mayr [May00]. Instead of (α, β)-sePRS or (α, β)-wPRS we juxtapose prefixes 'se-' or'w-' respectively with the acronym corresponding to the (α, β)-PRS class. For example, we use wBPP rather than $(1, P)$-wPRS.

The expressive power of a class is measured by the set of labelled transition systems that are definable (up to strong bisimulation equivalence [Mil89]) by the class. Details can be found in [KŘS04b, KŘS04a].

2.2 Logics and Studied Problems

In this paper we work with fragments of *unified system of branching-time logic* (UB) [BAPM83]. Formulae of UB have the following syntax:

$$\varphi \ ::= \ tt \ \mid \ \neg\varphi \ \mid \ \varphi_1 \wedge \varphi_2 \ \mid \ \langle a \rangle \varphi \ \mid \ \mathsf{EF}\varphi \ \mid \ \mathsf{EG}\varphi,$$

where $a \in Act$ is an action. Here, formulae are interpreted over states of sePRS systems. Validity of a formula φ in a state pt of a given sePRS system Δ, written $(\Delta, pt) \models \varphi$, is defined by induction on the structure of φ: tt is valid for all states; boolean operators have standard meaning; $(\Delta, pt) \models \langle a \rangle \varphi$ iff there is a state qt' such that $pt \xrightarrow{a} qt'$ and $(\Delta, qt') \models \varphi$; $(\Delta, pt) \models \mathsf{EF}\varphi$ iff there is a state qt' reachable from pt such that $(\Delta, qt') \models \varphi$; $(\Delta, pt) \models \mathsf{EG}\varphi$ iff there is a maximal (finite or infinite) transition sequence $p_1 t_1 \xrightarrow{a_1} p_2 t_2 \xrightarrow{a_2} p_3 t_3 \xrightarrow{a_3} \ldots$ such that $pt = p_1 t_1$ and all states in the sequence satisfy $p_i t_i \models \varphi$. We write $\Delta \models \varphi$ if φ is valid in the initial state $p_0 t_0$ of Δ. For each UB formula φ, $depth(\varphi)$ denotes a nesting depth of $\langle a \rangle$ operators in φ (see e.g. [Mil89] for this standard definition).

A UB formula φ is called

- an *EF formula* if it does not contain any EG operator;
- an *EG formula* if it does not contain any EF operator;
- a *Hennessy–Milner formula* (or *HM formula* for short) if it contains neither EG nor EF operators;
- a *simple formula* if it is an HM formula satisfying $depth(\varphi) = 1$.

In the following, we deal with six problems parametrized by a subclass of sePRS systems. Let Δ be a given system of the subclass considered. The problem to decide whether

- $\Delta \models \varphi$, where φ is a given EF formula, is called *decidability of EF logic*;
- $\Delta \models \mathsf{EF}\varphi$, where φ is a given HM formula, is called *reachability HM property*;
- $\Delta \models \mathsf{EF}\varphi$, where φ is a given simple formula, is called *reachability simple property*;
- $\Delta \models \varphi$, where φ is a given EG formula, is called *decidability of EG logic*;
- $\Delta \models \mathsf{EG}\varphi$, where φ is a given HM formula, is called *evitability HM property*;
- $\Delta \models \mathsf{EG}\varphi$, where φ is a given simple formula, is called *evitability simple property*.

3 Main Result

In this section, we study a *reachability HM property problem* for wPRS, i.e. the problem to decide whether a given wPRS Δ and a given HM formula φ satisfy $\Delta \models \mathsf{EF}\varphi$ or not. We prove that the problem is decidable. The proof reduces this problem to the *reachability problem* for wPRS, i.e. the problem to decide whether a given state of a given wPRS is reachable or not, which is decidable due to [KŘS04a].

Theorem 1 ([KŘS04a]). *The reachability problem for wPRS is decidable.*

For the rest of this section, let Δ be a fixed wPRS system, $D \notin \mathit{Const}(\Delta)$ be a fixed fresh process constant, and $\mathcal{C} = \mathit{Const}(\Delta) \cup \{D\}$. Further, let φ be a HM formula and $n = \mathit{depth}(\varphi)$. We assume that $n > 0$.

Definition 1. *A term t' is called* n-equivalent to *a state pt of Δ if and only if, for each HM formula ψ satisfying $\mathit{depth}(\psi) \leq n$, it holds:*

$$(\Delta, pt) \models \psi \iff (\Delta, pt') \models \psi$$

Our proof will proceed in two steps. In the first step we show that there exists a *finite* set T of terms such that, for each reachable state pt of Δ, the set T contains a term t' which is n-equivalent to pt. In the second step we enrich the system with rules allowing us to rewrite an arbitrary reachable state pt to a state $[p, t']\, D$, where the control state $[p, t']$ represents the original control state p and a term t' which is n-equivalent to pt. Finally, for each $p \in M(\Delta), t' \in T$ satisfying $(\Delta, pt') \models \varphi$ we add a rule $[p, t']\, D \overset{a}{\hookrightarrow} \mathit{acc}\, D$. Let us note that the validity of $(\Delta, pt') \models \varphi$ is decidable as wPRS systems are finitely branching. To sum up, φ is valid for some reachable state pt of Δ if and only if the state $\mathit{acc}\, D$ is reachable in the modified system.

First, we introduce some auxiliary terminology and notation. A nonempty proper subterm t' of a term t is called *idle* if t' is the right-hand-side component of some sequential composition in t (such that its left-hand-side component is nonempty), where sequential composition is considered to be left-associative. For example, a term $(X.Y.Z) \| (U.(V \| W))$ should be interpreted as $((X.Y).Z) \| (U.(V \| W))$ and its idle subterms are $Y, Z, V \| W$ but not $Y.Z$. By *IdleTerms* we denote a set of all idle terms occurring in the initial term or in terms on the right-hand sides of rewrite rules of Δ. Observe that each idle subterm of any reachable state of Δ is contained in *IdleTerms*.

We define a *length* of a term t, written $|t|$, as the number of all occurrences of process constants in the term. For example, $|X \| (X.Y) \| \varepsilon| = 3$. Further, for each $j \geq 0$, we define a set

$$\mathit{SmallTerms}(j) = \{t \mid t \text{ is a term over } \mathcal{C} \text{ and } 0 < |t| \leq j\}.$$

Definition 2. *Let $h > 0$ be an integer. We put $k = \max\{|t| \mid t \in \mathit{IdleTerms}\}$ and $H = h \cdot (h + k) \cdot |\mathit{SmallTerms}(h + k)|$. We define $\mathit{Rules}(h)$ to be the set of rewrite rules of three types (see the proof of Lemma 1 for their respective roles):*

$$
\begin{aligned}
&(1)\ \ p\,s'.D \overset{del}{\hookrightarrow} pD \quad \text{for all } p \in M(\Delta) \text{ and } s' \in \mathit{SmallTerms}(H),\\
&(2)\ \ p\,s^{h+1} \overset{del}{\hookrightarrow} p\,s^{h} \quad \text{for all } p \in M(\Delta) \text{ and } s \in \mathit{SmallTerms}(H),\\
&(3)\ \ p\,s'.s \overset{del}{\hookrightarrow} pD \quad \text{for all } p \in M(\Delta),\ s \in \mathit{IdleTerms}, \text{ and } s' \in \mathit{SmallTerms}(H) \setminus \mathit{SmallTerms}(h),
\end{aligned}
$$

where s^i denotes a parallel composition of i copies of term s.

Lemma 1. *For each $h > 0$ and for each reachable state pt of Δ it holds that $p\,t.D \xrightarrow{*}_{Rules(h)} pD$.*

Proof. As every rule in $Rules(h)$ has its right-hand side shorter than its left-hand side and an application of a rule in $Rules(h)$ cannot produce any new idle subterm, it is sufficient to prove that, for each $p \in M(\Delta)$ and each term t over $\mathcal{C}$ with all idle subterms in $\mathit{IdleTerms}$, there is a rule of $Rules(h)$ applicable to $p\,t.D$. We assume the contrary and derive a contradiction. Let $p \in M(\Delta)$ be a control state and t be a term of the minimal length such that t satisfies the preconditions and no rule of $Rules(h)$ is applicable to $p\,t.D$. Then $|t| > H$ as in the other case a rule of type (1) is applicable to $p\,t.D$. There are two cases:

- $t = u.v$ As $v \in \mathit{IdleTerms}$ we have $|v| \leq k$. Further, $|t| > H$ implies $|u| > H - k > h$. If $h < |u| \leq H$, then there is a rule of type (3) that can be applied to $p\,t.D$. Hence $|u| > H$. As no rule of $Rules(h)$ can be applied to $p\,t.D = p\,u.v.D$, no such rule can be applied to pu. The inequality $|u| > H$ gives us that a rule of type (1) is applicable to $p\,u.D$ if and only if it is applicable to pu. The same holds for rules of type (2) and (3) as well due to the shape of these rules and due to the fact that D does not occur in any term of $\mathit{IdleTerms}$. To sum up, no rule of $Rules(h)$ can be applied to $p\,u.D$ and thus u contradicts the minimality of t.
- $t = u \| v$ As '$\|$' is associative and commutative, it can be seen as an operator with an unbounded arity. Thus, t can be seen as a parallel composition of several components which are nonempty sequential terms. The length of each of these components is less than or equal to H; a component u satisfying $|u| > H$ would contradict the minimality of t using the same arguments as in the previous case. Further, as no rule of type (3) can be applied, the length of each component is at most $h + k$. Moreover, as rules of type (2) are not applicable, we have that the parallel composition contains at most h copies of each component. Hence, $|t| \leq h \cdot (h + k) \cdot |\mathit{SmallTerms}(h + k)| = H$. This contradicts the relation $|t| > H$. □

Definition 3. *Let l be the maximal length of a left-hand-side term of a rule in Δ. Lemma 1 implies that, for each reachable state pt of Δ, there exists a transition sequence $p\,t.D \xrightarrow{*}_{Rules(nl)} pD$. By $\mathit{MultiSet}_{nl}(pt)$ (or just $\mathit{MultiSet}(pt)$ if no confusion can arise) we denote a multiset containing exactly all the subterms that are rewritten during this transition sequence and correspond to a subterm s' of rewrite rules of types (1) and (3). Further, for each multiset of terms $S = \{t_1, t_2, \ldots, t_j\}$, we define its* characteristic term *$t_S = (t_1.D) \| (t_2.D) \| \ldots \| (t_j.D)$.*

Lemma 2. *Let pt be a reachable state of Δ. Then $t_{MultiSet(pt)}$ is n-equivalent to pt.*

Proof. Let us fix a transition sequence $p\,t.D \xrightarrow{*}_{Rules(nl)} pD$ and the corresponding multiset $MultiSet(pt)$. The proof proceeds by induction on the number of transition steps in the transition sequence.

Let S_i denote a part of $MultiSet(pt)$ obtained in the first i transition steps and pu_i be the state reached by these steps. It is sufficient to prove that, for each i, $u_i \| t_{S_i}$ is n-equivalent to pt. We note that D cannot be rewritten by any rewrite rule of Δ – it is used to prevent unwanted rewriting.

The basic step is trivial as $u_0 = t$, $S_0 = \emptyset$, and thus $u_i \| t_{S_i} = t \| \varepsilon$. Now we assume that $u_i \| t_{S_i}$ is n-equivalent to pt and we prove that the same holds for $u_{i+1} \| t_{S_{i+1}}$. Let l be the maximal length of a left-hand-side term of a rule in Δ. There are three cases reflecting the type of the rewrite rule applied in a transition step $pu_i \xrightarrow{del}_{Rules(nl)} pu_{i+1}$:

type (1) We note that no rule in $Rules(nl)$ can introduce D on the right-hand side of a sequential composition. Thus, a rule $p\,s'.D \overset{del}{\hookrightarrow} pD$ of type (1) is applicable to pu_i iff $u_i = s'.D$. Therefore, $u_{i+1} = D$, $S_{i+1} = S_i \cup \{s'\}$, and $u_{i+1} \| t_{S_{i+1}} = D \| (s'.D) \| t_{S_i} = D \| u_i \| t_{S_i}$. As $u_i \| t_{S_i}$ is n-equivalent to pt, it is obvious that so is $u_{i+1} \| t_{S_{i+1}}$.

type (2) Let ψ be a HM formula such that $depth(\psi) \leq n$. Then its validity in a state depends only on the first n successive transitions performable from the state. At most nl process constants of the term t can be rewritten during n successive steps. Hence, at most nl parallel components can be rewritten during these steps. Thus, reducing of the number of identical parallel components from $nl + 1$ to nl does not affect the validity of ψ. To sum up, $u_{i+1} \| t_{S_{i+1}} = u_{i+1} \| t_{S_i}$ is n-equivalent to pt.

type (3) The term s' occurring in the applied rule satisfies $|s'| > nl$. Hence, the part of the term t corresponding to the subterm s of the rule is "too far" to be rewritten in the first n steps of any transition sequence. The term $s'.s$ in u_i is replaced by D in u_{i+1}. It is easy to see that $u_{i+1} \| t_{S_{i+1}} = u_{i+1} \| (s'.D) \| t_{S_i}$ is n-equivalent to pt. □

Given a multiset of terms S, by $S{\downarrow}n$ we denote the largest subset of S containing at most n copies of each element. One can readily confirm that a characteristic term t_S is n-equivalent to some state of Δ if and only if $t_{S\downarrow n}$ is n-equivalent to this state.

To sum up, for each reachable state pt of Δ, we can construct a multiset $MultiSet(pt){\downarrow}n$ such that its characteristic term $t_{MultiSet(pt)\downarrow n}$ is n-equivalent to pt. Moreover, there is a bound on the size of each such a multiset which depends on Δ and n only. More precisely, such a multiset contains at most n copies of terms $s' \in SmallTerms(nl \cdot (nl + k) \cdot |SmallTerms(nl + k)|)$, where l is the maximal length of a left-hand-side term of a rule in Δ and k the maximal length of a term in $IdleTerms$. We now present the reduction of the reachability HM property problem for wPRS to the reachability problem for wPRS.

Lemma 3. *Let Δ be a wPRS system and φ be a Hennessy–Milner formula. Then we can construct a wPRS Δ' with a state $acc\,D$ such that*

$$\Delta \models \mathsf{EF}\varphi \iff acc\,D \text{ is reachable in } \Delta'.$$

Proof. Let n, D, $\mathcal{C}$, $IdleTerms$, $SmallTerms(j)$, and $MultiSet(pt)$ have the same meanings as above.

Let k be the maximal length of a term in $IdleTerms$, l be the maximal length of a left-hand-side term in any rule from Δ, and $H = nl{\cdot}(nl{+}k){\cdot}|SmallTerms(nl{+}k)|$. Further, let $\mathcal{MS}$ be a set of all multisets containing at most n copies of each term $s' \in SmallTerms(H)$.

The system Δ' uses control states of the original system, a distinguished control state $acc \notin M(\Delta)$, and control states of the form (p, S) where $p \in M(\Delta)$ and $S \in \mathcal{MS}$.

Let p_0t_0 be the initial state of Δ. Then Δ' has the initial state $p_0t_0.D$ and the following rules, where p and S range over $M(\Delta)$ and $\mathcal{MS}$ respectively. We omit labels as they are not relevant.

(1)	$pt \hookrightarrow qt'$	for all $(pt \overset{a}{\hookrightarrow} qt') \in \Delta$
(2)	$pX \hookrightarrow (p, \emptyset)\,X$	for all $X \in \mathcal{C}$
(3)	$(p, S)\,s'.D \hookrightarrow (p, (S \cup \{s'\})\downarrow n)\,D$	for all $s' \in SmallTerms(H)$
(4)	$(p, S)\,s^{nl+1} \hookrightarrow (p, S)\,s^{nl}$	for all $s \in SmallTerms(H)$
(5)	$(p, S)\,s'.s \hookrightarrow (p, (S \cup \{s'\})\downarrow n)\,D$	for all $s \in IdleTerms$ and $s' \in SmallTerms(H) \setminus SmallTerms(nl)$
(6)	$(p, S)\,D \hookrightarrow acc\,D$	whenever $(\Delta, pt_S) \models \varphi$

Intuitively, the rules of type (1) mimic the behaviour of Δ and allow Δ' to reach a state $p\,t.D$ if and only if pt is a reachable state of Δ. A rule of type (2) stops this mimic phase and starts a checking phase where only rules of types (3)–(6) are applicable. The rules of types (3), (4), and (5) correspond to the rules of type (1), (2), and (3) in $Rules(nl)$, respectively. Let $p\,t.D$ be a final state reached in the mimic phase. The rules of types (3)–(5) allow us to rewrite this state to the state $(p, MultiSet(pt)\downarrow n)\,D$. Finally, the control state $(p, MultiSet(pt)\downarrow n)$ can be changed to acc using a rule of type (6) if and only if $(\Delta, t_{MultiSet(pt)\downarrow n}) \models \varphi$. As $t_{MultiSet(pt)\downarrow n}$ is n-equivalent to pt, the control state can be changed to acc if and only if $(\Delta, pt) \models \varphi$. □

The following theorem is an immediate corollary of Lemma 3 and Theorem 1.

Theorem 2. *The reachability HM property problem is decidable for wPRS.*

4 Related Results

An interesting corollary of Theorem 2 arises in connection with one of the results of [JKM01].

Theorem 3 ([JKM01], Theorem 22). *If the model checking problem for simple EF formulae (i.e. reachability HM property problem) is decidable in a class K of transition systems, then strong bisimilarity is decidable between processes of K and finite-state ones.*

A combination of Theorem 2 and Theorem 3 yields the following corollary.

Theorem 4. *Strong bisimilarity is decidable between wPRS systems and finite-state ones.*

Remark 1. Theorem 2 also implies that reachability simple property problem is decidable for PRS. This result has been previously presented in [May98] under the name *reachable property* problem. However, the proof given there contains a nontrivial mistake which was not fixed in subsequent papers [MR98, May00]. The weak point is the proof showing a transformation of an arbitrary PRS onto a PRS in normal form. Considering a PRS $\Delta = (\{A\|(B.C) \stackrel{a}{\hookrightarrow} A\|(B.C)\}, A\|(B.C))$ that does not model a formula $\mathsf{EF}(\neg\langle a\rangle tt)$, one receives a transformed PRS in normal form that models this formula.

Remark 2. It is known that (full) EF logic is undecidable for PN [Esp94]. An inspection of the proof given in [Esp97] shows that this undecidability result is valid even for seBPP class (also known as *multiset automata*, MSA).

Remark 3. Esparza and Kiehn have proved that EG logic is undecidable for (deterministic) BPP [EK95]. In Appendix A we describe a modification of their proof showing that for (deterministic) BPP even the evitability simple property problem is undecidable.

5 Conclusion

In the paper we have shown that given any wPRS system Δ and any Hennessy–Milner formula φ, one can decide whether there is a state s of Δ reachable from the initial state of Δ such that s satisfies φ. Using Theorem 22 of [JKM01], our result implies that strong bisimilarity between wPRS and finite-state systems is decidable. Decidability of the same problem for some of the wPRS subclasses, namely PAN and PRS, has been formulated as an open question, see e.g. [Srb02].

The following table describes the current state of (un)decidability results regarding the six problems defined at the end of Section 2 for the classes of PRS hierarchy and their extended counterparts. The results established by this paper are typeset in bold.

problem	decidable for	undecidable for
decidability of EF logic	PAD [May98]	**seBPP**
reachability HM property	**wPRS**	sePA
reachability simple property	**wPRS**	sePA
decidability of EG logic	PDA [MS85, Cau92]	BPP [EK95]
evitability HM property	PDA [MS85, Cau92]	BPP [EK95]
evitability simple property	PDA [MS85, Cau92]	**BPP**

To sum up, the situation with (un)decidability of these six problems for all the considered classes is now clear with one exception: decidability of EF logic remains open for classes wBPP, wPA, and wPAD.

Regarding other decidability questions we note that the BPP class and its extensions form a strict (sub)hierarchy with respect to bisimulation,

$$\text{BPP} \subsetneq \text{wBPP} \subsetneq \text{MSA} \subsetneq \text{PN},$$

which is decidable (even PSPACE-complete [Jan03]) for BPP processes and undecidable for MSA ([HM01] using the techniques of [Jan95]). Decidability of bisimilarity remains open for the wBPP class and is a subject of our further research. We are motivated by the fact the strictness of the left-most inclusion can be proved (but is not shown here) even for language equivalence. The strictness of inclusion between wBPP and MSA on the language equivalence level is just our conjecture.

Acknowledgment. We thank Antonín Kučera for valuable suggestions and pointers. Authors have been partially supported as follows: M. Křetínský by the Grant Agency of the Czech Republic, grant No. 201/03/1161, V. Řehák by the research centre "Institute for Theoretical Computer Science (ITI)", project No. 1M0021620808, and J. Strejček by the Academy of Sciences of the Czech Republic, grant No. 1ET408050503.

References

[BAPM83] M. Ben-Ari, A. Pnueli, and Z. Manna. The temporal logic of branching time. *Acta Informatica*, 20(3):207–226, 1983.

[BCMS01] O. Burkart, D. Caucal, F. Moller, and B. Steffen. Verification on infinite structures. In *Handbook of Process Algebra*, pages 545–623. Elsevier, 2001.

[BCS96] O. Burkart, D. Caucal, and B. Steffen. Bisimulation collapse and the process taxonomy. In *Proc. of CONCUR'96*, volume 1119 of *LNCS*, pages 247–262. Springer, 1996.

[BEH95] A. Bouajjani, R. Echahed, and P. Habermehl. On the verification problem of nonregular properties for nonregular processes. In *Proc. of LICS'95*. IEEE, 1995.

[Cau92] D. Caucal. On the regular structure of prefix rewriting. *Theor. Comput. Sci.*, 106:61–86, 1992.

[EK95] J. Esparza and A. Kiehn. On the model checking problem for branching time logics and basic parallel processes. In *CAV*, volume 939 of *LNCS*, pages 353–366. Springer, 1995.

[Esp94] J. Esparza. On the decidability of model checking for several mu-calculi and petri nets. In *CAAP*, volume 787 of *LNCS*, pages 115–129. Springer, 1994.

[Esp97] J. Esparza. Decidability of model checking for infinite-state concurrent systems. *Acta Informatica*, 34(2):85–107, 1997.

[Esp02] J. Esparza. Grammars as processes. In *Formal and Natural Computing*, volume 2300 of *LNCS*. Springer, 2002.

[HM01] Y. Hirshfeld and F. Moller. Pushdown automata, multiset automata, and petri nets. *Theor. Comput. Sci.*, 256(1-2):3–21, 2001.

[Jan95] P. Jančar. Undecidability of bisimilarity for Petri nets and some related problems. *Theor. Comput. Sci.*, 148(2):281–301, 1995.

[Jan03] P. Jančar. Strong bisimilarity on basic parallel processes is PSPACE-complete. In *Proc. of 18th IEEE Symposium on Logic in Computer Science (LICS'03)*, pages 218–227. IEEE Computer Society, 2003.

[JKM01] P. Jančar, A. Kučera, and R. Mayr. Deciding bisimulation-like equivalences with finite-state processes. *Theor. Comput. Sci.*, 258:409–433, 2001.

[JM95] P. Jancar and F. Moller. Checking regular properties of petri nets. In *CONCUR*, volume 962 of *LNCS*, pages 348–362. Springer, 1995.

[KJ02] A. Kučera and P. Jančar. Equivalence-checking with infinite-state systems: Techniques and results. In *Proc. SOFSEM'2002*, volume 2540 of *LNCS*. Springer, 2002.

[KŘS04a] M. Křetínský, V. Řehák, and J. Strejček. Extended process rewrite systems: Expressiveness and reachability. In *Proceedings of CONCUR'04*, volume 3170 of *LNCS*, pages 355–370. Springer, 2004.

[KŘS04b] M. Křetínský, V. Řehák, and J. Strejček. On extensions of process rewrite systems: Rewrite systems with weak finite-state unit. In *Proceedings of INFINITY'03*, volume 98 of *ENTCS*, pages 75–88. Elsevier, 2004.

[KS04] A. Kučera and Ph. Schnoebelen. A general approach to comparing infinite-state systems with their finite-state specifications. In *CONCUR*, volume 3170 of *LNCS*, pages 371–386. Springer, 2004.

[May98] R. Mayr. *Decidability and Complexity of Model Checking Problems for Infinite-State Systems*. PhD thesis, Technische Universität München, 1998.

[May00] R. Mayr. Process rewrite systems. *Information and Computation*, 156(1):264–286, 2000.

[Mil89] R. Milner. *Communication and Concurrency*. Prentice-Hall, 1989.

[Mol96] F. Moller. Infinite results. In *Proc. of CONCUR'96*, volume 1119 of *LNCS*, pages 195–216. Springer, 1996.

[MR98] R. Mayr and M. Rusinowitch. Reachability is decidable for ground AC rewrite systems. In *Proceedings of INFINITY'98 workshop*, 1998.

[MS85] D. Muller and P. Schupp. The theory of ends, pushdown automata, and second-order logic. *Theor. Comput. Sci.*, 37:51–75, 1985.

[MSS92] D. Muller, A. Saoudi, and P. Schupp. Alternating automata, the weak monadic theory of trees and its complexity. *Theor. Comput. Sci.*, 97(1–2):233–244, 1992.

[Srb02] J. Srba. Roadmap of infinite results. *EATCS Bulletin*, (78):163–175, 2002. `http://www.brics.dk/~srba/roadmap/`.

A Evitability Simple Property for Deterministic BPP

In this section, we show how to strengthen the result of undecidability the EG logic for BPP, a proof of which has been given by Esparza and Kiehn in [EK95]. As we just describe the necessary changes to be done within the proof, we use the same notation as introduced in [EK95]. The original proof is done by a reduction from the halting problem of a Minsky counter machine. A quick inspection of the reduction shows that it demonstrates undecidability of the *inevitability HM*

property problem for the class of deterministic BPP systems. We note that it is not a proof of undecidability for the *inevitability simple property problem* due to the following reason. In the definition of $\widehat{EN}(a_1, \ldots, a_k)$, there is a subformula

$$\bigwedge_{i:=1}^{k} \neg\exists(a_i)EN(a_i) \quad \text{corresponding to} \quad \bigwedge_{i=1}^{k} \neg\langle a_i\rangle\langle a_i\rangle tt \quad \text{in our notation}$$

which expresses that no sequence $a_i a_i$ is enabled. Omitting this subformula from $\widehat{EN}(a_1, \ldots, a_k)$, the construction produces a simple property formula.

In what follows, we present some other changes to be done within the construction in orded to keep its correctness for the case of the simple property formula as well. In other words, we prove than even the inevitability *simple* property problem remains undecidable for the deterministic BPP systems.

The following definitions of $\mathtt{SM}$, $\mathtt{M}$, and $\mathtt{C_j}$ are the same as in [EK95]:

$$\mathtt{SM} \stackrel{\text{def}}{=} (\mathtt{SQ_1} \| \ldots \| \mathtt{SQ_{n+1}}) \qquad \mathtt{M} \stackrel{\text{def}}{=} \mathtt{SM} \| \mathtt{Q_0} \qquad \mathtt{C_j} \stackrel{\text{def}}{=} \mathtt{dec^1_j} \cdot \mathtt{dec^2_j} \cdot \mathtt{dec^3_j} \cdot \mathbf{0}$$

Without loss of generality, we assume that there is no self loop in the counter machine $\mathcal{M}$ (i.e., $k \neq i \neq k'$, for each transition rule of $\mathcal{M}$). Hence, it is not necessary to create a new parallel instance of a process constant $\mathtt{Q_i}$ from $\mathtt{SQ_i}$ as far as the rewriting on the existing instance of $\mathtt{Q_i}$ is not finished. In the following, we reformulate the definitions of $\mathtt{SQ_i}$ and $\mathtt{Q_i}$ to prevent sequences of the form $\mathtt{out^1_i out^1_i}$ or $\mathtt{out^2_i out^2_i}$.

The halting state definition is reformulated as follows.

$$\mathtt{SQ_{n+1}} \stackrel{\text{def}}{=} \mathtt{in^1_{n+1}} \cdot \mathtt{Q_{n+1}} \qquad\qquad \mathtt{Q_{n+1}} \stackrel{\text{def}}{=} \mathtt{halt} \cdot \mathtt{SQ_{n+1}}$$

A state q_i of type II is modelled as follows.

$$\mathtt{SQ_i} \stackrel{\text{def}}{=} \mathtt{in^1_i} \cdot \mathtt{Q_i} \qquad\qquad \mathtt{Q_i} \stackrel{\text{def}}{=} \mathtt{out^1_i} \cdot \mathtt{out^2_i} \cdot \mathtt{SQ_i}$$

A state q_i of type I has to proceed to the state q_k. To prevent multiple occurrences of the process constant $\mathtt{Q_k}$, we use the same technique as in the case of states of type II. Hence, $\mathtt{SQ_i}$ and $\mathtt{Q_i}$ are modelled as

$$\mathtt{SQ_i} \stackrel{\text{def}}{=} \mathtt{in^1_i} \cdot \mathtt{Q_i} \qquad\qquad \mathtt{Q_i} \stackrel{\text{def}}{=} \mathtt{out^1_i} \cdot \mathtt{out^2_i} \cdot (\mathtt{SQ_i} \| \mathtt{C_j})$$

and we add the following disjunct to the formula ϕ_h to guarantee a move to the state q_k in an honest run.

$$\widehat{EN}(\mathtt{out^1_i}) \vee \widehat{EN}(\mathtt{out^2_i}) \vee \widehat{EN}(\mathtt{out^2_i}, \mathtt{out^1_k}) \vee \widehat{EN}(\mathtt{out^1_k})$$

Hence, the multiple enabling of $\mathtt{out^1_i}$ and $\mathtt{out^2_i}$ is omitted by the construction. It remains to focus on the situation for $\mathtt{dec^2_i}$ and $\mathtt{dec^3_i}$. As the states where both $\mathtt{dec^2_i}$ and $\mathtt{dec^3_i}$ are enabled do not satisfy ϕ_h, each state satisfying $\exists(\mathtt{dec^2_i})EN(\mathtt{dec^2_i})$ has no continuation to make a honest run and each state satisfying $\exists(\mathtt{dec^3_i})EN(\mathtt{dec^3_i})$ is unreachable in any honest run.