On Decidability of LTL Model Checking for Process Rewrite Systems

Laura Bozzelli[1], Mojmír Křetínský[2], Vojtěch Řehák[2], and Jan Strejček[2]

[1] Dipartimento di Matematica e Apllicazioni, Università degli Studi di Napoli "Federico II", Via Cintia, 80126 Napoli, Italy
laura.bozzelli@dma.unina.it
[2] Faculty of Informatics, Masaryk University, Botanická 68a, 60200 Brno, Czech Republic
{kretinsky, rehak, strecek}@fi.muni.cz

Abstract. We establish a decidability boundary of the model checking problem for infinite-state systems defined by *Process Rewrite Systems* (PRS) or *weakly extended Process Rewrite Systems* (wPRS), and properties described by basic fragments of action-based *Linear Temporal Logic* (LTL). It is known that the problem for general LTL properties is decidable for Petri nets and for pushdown processes, while it is undecidable for PA processes. As our main result, we show that the problem is decidable for wPRS if we consider properties defined by formulae with only modalities *strict eventually* and *strict always*. Moreover, we show that the problem remains undecidable for PA processes even with respect to the LTL fragment with the only modality *until* or the fragment with modalities *next* and *infinitely often*.

1 Introduction

Automatic verification of current software systems often needs to model them as infinite-state systems. One of the most powerful formalisms for description of infinite-state systems (except formalisms with Turing power for which nearly all interesting verification problems are undecidable) is called *Process Rewrite Systems* (PRS) [May00]. The PRS framework, based on term rewriting, subsumes many formalisms studied in the context of formal verification, e.g. *Petri nets* (PN), *pushdown processes* (PDA), and process algebras like BPA, BPP, or PA. PRS can be adopted as a formal model for programs with recursive procedures and restricted forms of dynamic creation and synchronization of concurrent processes. A substantial advantage of PRS is that some important verification problems are decidable for the whole PRS class. In particular, Mayr [May00] proved that the *reachability problem* (whether a given state is reachable) and the *reachable property problem* (whether there is a reachable state where some given actions are enabled and some given actions are disabled) are decidable for PRS.

In [KŘS04b], we have presented *weakly extended PRS* (wPRS), where a finite-state control unit with self-loops as the only loops is added to the standard PRS formalism (addition of a general finite-state control unit makes PRS Turing powerful). This control unit enriches PRS by abilities to model a bounded number of arbitrary communication events and global variables whose values are changed only a bounded number of times

S. Arun-Kumar and N. Garg (Eds.): FSTTCS 2006, LNCS 4337, pp. 248–259, 2006.

during any computation. We have proved that the reachability problem remains decidable for wPRS [KŘS04a] and that the problem called *reachability Hennessy–Milner property* (whether there is a reachable state satisfying a given Hennessy–Milner formula) is decidable for wPRS as well [KŘS05]. The hierarchy of all PRS and wPRS classes is depicted in Figure 1.

Concerning the model checking problem, a broad overview of (un)decidability results for subclasses of PRS and various temporal logics can be found in [May98]. Here we focus exclusively on (future) *Linear Temporal Logic* (LTL). It is known that LTL model checking of PDA is EXPTIME-complete [BEM97]. LTL model checking of PN is also decidable, but at least as hard as the reachability problem for PN [Esp94] (the reachability problem is EXPSPACE-hard [May84, Lip76] and no primitive recursive upper bound is known). If we consider only infinite runs, then the problem for PN is EXPSPACE-complete [Hab97, May98].

Conversely, LTL model checking is undecidable for all classes subsuming PA [BH96, May98]. So far, there are only two positive results for these classes. Bouajjani and Habermehl [BH96] have identified a fragment called *simple PLTL*$_\Box$ for which model checking of infinite runs is decidable for PA (strictly speaking, simple PLTL$_\Box$ is not a fragment of LTL as it can express also some non-regular properties, while LTL cannot). Only recently, we have demonstrated that model checking of infinite runs is decidable for PRS and the fragment of LTL capturing exactly fairness properties [Boz05].

Our contribution: This paper completely locates the decidability boundary of the model checking problem for all subclasses of PRS (and wPRS) and all *basic LTL fragments*, where a basic LTL fragment is a set of all formulae containing only a given subset of standard modalities. The boundary is depicted in Figure 2. To locate the boundary, we show the following results.

1. We introduce a new LTL fragment $\mathcal{A}$ and prove that every formula of the basic fragment LTL$(\mathsf{F}_s, \mathsf{G}_s)$ (i.e. the fragment with modalities *strict eventually* and *strict always* only) can be effectively translated into $\mathcal{A}$. As LTL$(\mathsf{F}_s, \mathsf{G}_s)$ is closed under negation, we can also translate LTL$(\mathsf{F}_s, \mathsf{G}_s)$ formulae into negated formulae of $\mathcal{A}$.
2. We show that model checking (of both finite and infinite runs) of wPRS against negated formulae of $\mathcal{A}$ is decidable. The proof employs our results presented in [Boz05, KŘS04a, KŘS05] to reduce the problem to LTL model checking for PDA and PN. Thus we get decidability of model checking for wPRS against LTL$(\mathsf{F}_s, \mathsf{G}_s)$. Note that LTL$(\mathsf{F}_s, \mathsf{G}_s)$ is strictly more expressive than the *Lamport logic* (i.e. the basic fragment with modalities *eventually* and *always*), which is again strictly more expressive than the mentioned fragment of fairness properties and also than the *regular* part of simple PLTL$_\Box$.
3. We demonstrate that the model checking problem remains undecidable for PA even if we consider the basic fragment with modality *until* or the basic fragment with modalities *next* and *infinitely often* (which is strictly less expressive than the one with *next* and *eventually*).

The paper is organized as follows. The following section recalls basic definitions. Sections 3, 4, and 5 correspond, respectively, to the three results listed above. The last section discuss other potential applications of our results and it contains an open

question driving our future research. Proofs are only sketched due to space constraints. Full proofs can be found in [BKŘS06].

2 Preliminaries

2.1 PRS and Its Extensions

Let $Const = \{X, \ldots\}$ be a set of *process constants*. The set of *process terms* t is defined by the abstract syntax $t ::= \varepsilon \mid X \mid t.t \mid t \| t$, where ε is the *empty term*, $X \in Const$, 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, 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*, $\leq$ be a partial ordering on this set, and $Act = \{a, b, c, \ldots\}$ be a set of *actions*. Let $\alpha, \beta \in \{1, S, P, G\}$ be classes of process terms such that $\alpha \subseteq \beta$. An (α, β)-*wPRS* (*weakly extended process rewrite system*) Δ is a tuple (R, p_0, X_0), where

- R is a finite set of *rewrite rules* of the form $(p, t_1) \overset{a}{\hookrightarrow} (q, t_2)$, where $t_1 \in \alpha$, $t_1 \neq \varepsilon$, $t_2 \in \beta$, $a \in Act$, and $p, q \in M$ are control states satisfying $p \leq q$,
- the pair $(p_0, X_0) \in M \times Const$ forms the distinguished *initial state*.

By $Act(\Delta)$, $Const(\Delta)$, and $M(\Delta)$ we denote the sets of actions, process constants, and control states occurring in the rewrite rules or the initial state of Δ, respectively.

An (α, β)-wPRS $\Delta = (R, p_0, X_0)$ induces a labelled transition system, whose states 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 $\longrightarrow_\Delta$ is the least relation satisfying the following inference rules:

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

Sometimes we write $\longrightarrow$ instead of $\longrightarrow_\Delta$ if Δ is clear from the context. The transition relation can be extended to finite words over Act in a standard way. To shorten our notation we write pt in lieu of (p,t). A state pt is called *terminal*, written $pt \not\longrightarrow_\Delta$, if there is no state $p't'$ and action a such that $pt \overset{a}{\longrightarrow}_\Delta p't'$. In this paper we always consider only systems where the initial state is not terminal. A (finite or infinite) sequence

$$\sigma = p_1 t_1 \overset{a_1}{\longrightarrow}_\Delta p_2 t_2 \overset{a_2}{\longrightarrow}_\Delta \cdots \overset{a_n}{\longrightarrow}_\Delta p_{n+1} t_{n+1} \left(\overset{a_{n+1}}{\longrightarrow}_\Delta \cdots \right)$$

is called *derivation over the word* $u = a_1 a_2 \ldots a_n (a_{n+1} \ldots)$ *in* Δ. Finite derivations are also denoted as $p_1 t_1 \overset{u}{\longrightarrow}_\Delta p_{n+1} t_{n+1}$, infinite as $p_1 t_1 \overset{u}{\longrightarrow}_\Delta$. A derivation in Δ is called

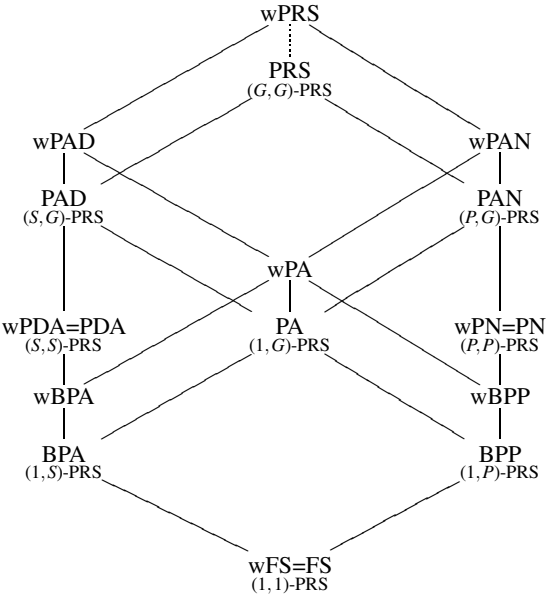

Fig. 1. The hierarchy of PRS and wPRS subclasses

a *run of* Δ if it starts in the initial state p_0X_0 and it is either infinite, or its last state is terminal. Further, $L(\Delta)$ denotes the set of words u such that there is a run of Δ over u.

An (α,β)-wPRS Δ where $M(\Delta)$ is a singleton is called (α,β)-*PRS* (*process rewrite system*) [May00]. In such systems we omit the single control state from rules and states.

Some classes of (α,β)-PRS correspond to widely known models, namely *finite-state systems* (FS), *basic process algebras* (BPA), *basic parallel processes* (BPP), *process algebras* (PA), *pushdown processes* (PDA), and *Petri nets* (PN). The other classes have been named as PAD, PAN, and PRS. The relations between (α,β)-PRS and the mentioned formalisms and names are indicated in Figure 1. Instead of (α,β)-wPRS we juxtapose the prefix 'w-' with the acronym corresponding to the (α,β)-PRS class. For example, we use wBPA rather than $(1,S)$-wPRS. Figure 1 shows the expressiveness hierarchy of all considered classes, where expressive power of a class is measured by the set of transition systems that are definable (up to the strong bisimulation equivalence [Mil89]) by the class. This hierarchy is strict, with a potential exception concerning the classes wPRS and PRS, where the strictness is just our conjecture. For details see [KŘS04b, KŘS04a].

For technical reasons, we define a normal form of wPRS systems. A rewrite rule is *parallel* or *sequential* if it has one of the following forms:

$$\textbf{Parallel rules}: \quad pX_1\|X_2\|\ldots\|X_n \overset{a}{\hookrightarrow} qY_1\|Y_2\|\ldots\|Y_m$$
$$\textbf{Sequential rules}: \quad pX \overset{a}{\hookrightarrow} qY.Z \qquad pX.Y \overset{a}{\hookrightarrow} qZ \qquad pX \overset{a}{\hookrightarrow} qY \qquad pX \overset{a}{\hookrightarrow} q\varepsilon$$

where $X,Y,X_i,Y_j,Z \in Const$, $p,q \in M$, $n>0$, $m \geq 0$, and $a \in Act$. A rule is called *trivial* if it is both parallel and sequential (i.e. it has the form $pX \overset{a}{\hookrightarrow} qY$ or $pX \overset{a}{\hookrightarrow} q\varepsilon$). A wPRS Δ is in *normal form* if it has only parallel and sequential rewrite rules.

2.2 Linear Temporal Logic (LTL) and Studied Problems

The syntax of *Linear Temporal Logic* (LTL) [Pnu77] is defined as follows

$$\varphi ::= tt \mid a \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathsf{X}\varphi \mid \varphi \mathsf{U} \varphi,$$

where a ranges over Act, X is called *next*, and U is called *until*. The logic is interpreted over infinite as well as nonempty finite words of actions. Given a word $u = u(0)u(1)u(2)\ldots \in Act^* \cup Act^\omega$, $|u|$ denotes the length of the word (we set $|u| = \infty$ if u is infinite). For all $0 \leq i < |u|$, by u_i we denote the i^{th} suffix of u, i.e. $u_i = u(i)u(i+1)\ldots$.

The semantics of LTL formulae is defined inductively as follows:

$$\begin{array}{lll}
u \models tt & & \\
u \models a & \text{iff} & u(0) = a \\
u \models \neg\varphi & \text{iff} & u \not\models \varphi \\
u \models \varphi_1 \wedge \varphi_2 & \text{iff} & u \models \varphi_1 \text{ and } u \models \varphi_2 \\
u \models \mathsf{X}\varphi & \text{iff} & |u| > 1 \text{ and } u_1 \models \varphi \\
u \models \varphi_1 \mathsf{U} \varphi_2 & \text{iff} & \exists 0 \leq i < |u| \,.\, (u_i \models \varphi_2 \text{ and } \forall 0 \leq j < i \,.\, u_j \models \varphi_1)
\end{array}$$

We say that a nonempty word u *satisfies* φ whenever $u \models \varphi$. Given a set of words L, we write $L \models \varphi$ if $u \models \varphi$ holds for all $u \in L$. We say that a derivation (or run) σ over a word u satisfies φ, written $\sigma \models \varphi$, whenever $u \models \varphi$.

Moreover, we define the following modalities: $\mathsf{F}\varphi$ (*eventually*) standing for $tt\,\mathsf{U}\,\varphi$, $\mathsf{G}\varphi$ (*always*) standing for $\neg\mathsf{F}\neg\varphi$, $\mathsf{F_s}\varphi$ (*strict eventually*) standing for $\mathsf{XF}\varphi$, $\mathsf{G_s}\varphi$ (*strict always*) standing for $\neg\mathsf{F_s}\neg\varphi$, $\overset{\infty}{\mathsf{F}}\varphi$ (*infinitely often*) standing for $\mathsf{GF}\varphi$, $\overset{\infty}{\mathsf{G}}\varphi$ (*almost always*) standing for $\neg\overset{\infty}{\mathsf{F}}\neg\varphi$. Note that $\mathsf{F}\varphi$ is equivalent to $\varphi \vee \mathsf{F_s}\varphi$ but $\mathsf{F_s}\varphi$ cannot be expressed with F as the only modality. Thus $\mathsf{F_s}$ is "stronger" than F. The relation between $\mathsf{G_s}$ and G is similar.

For a set $\{O_1, \ldots, O_n\}$ of modalities, $\mathrm{LTL}(O_1, \ldots, O_n)$ denotes the LTL fragment containing all formulae with modalities $O_1, \ldots, O_n$ only. Such a fragment is called *basic*. Figure 2 shows an expressiveness hierarchy of all studied basic LTL fragments. Indeed, every basic LTL fragment using standard[1] future modalities is equivalent to one of the fragments in the hierarchy, where equivalence between fragments means that every formula of one fragment can be effectively translated into a semantically equivalent formula of the other fragment and vice versa. For example, $\mathrm{LTL}(\mathsf{F_s}, \mathsf{G_s}) \equiv \mathrm{LTL}(\mathsf{F_s})$. Further, the hierarchy is strict. For detailed information about expressiveness of future LTL modalities and LTL fragments we refer to [Str04].

Let $\mathcal{F}$ be an LTL fragment and $\mathcal{C}$ be a class of wPRS systems. The *model checking problem* for $\mathcal{F}$ and $\mathcal{C}$ is to decide whether a given formula $\varphi \in \mathcal{F}$ and a given system $\Delta \in \mathcal{C}$ satisfies $L(\Delta) \models \varphi$. We also mention the problem called *model checking of infinite runs*, where $L(\Delta) \cap Act^\omega \models \varphi$ is examined.

[1] By standard modalities we mean the ones defined in this paper and also other commonly used modalities like *strict until*, *release*, *weak until*, etc. However, it is well possible that one can define a new modality such that there is a basic fragment not equivalent to any of the fragments in the hierarchy.

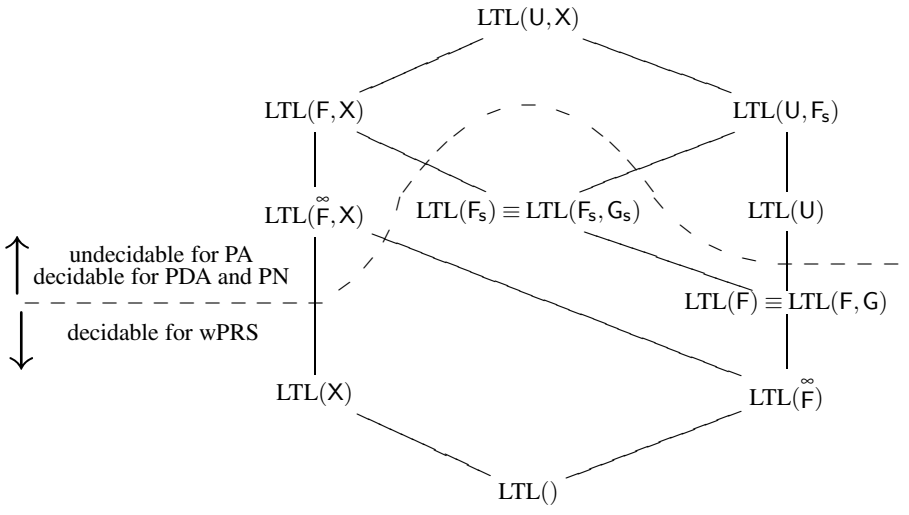

Fig. 2. The hierarchy of basic fragments with model checking decidability boundary

3 Fragment $\mathcal{A}$ and Translation of LTL$(\mathsf{F_s},\mathsf{G_s})$ into $\mathcal{A}$

The $\mathcal{A}$ fragment consists of finite disjunctions of α-*formulae* defined as follows.

Recall that LTL() denotes the fragment of formulae without any modality, i.e. boolean combinations of actions. In the following we use $\varphi_1 \mathsf{U}_+ \varphi_2$ to abbreviate $\varphi_1 \wedge \mathsf{X}(\varphi_1 \mathsf{U} \varphi_2)$. Let $\delta = \theta_1 O_1 \theta_2 O_2 \ldots \theta_n O_n \theta_{n+1}$, where $n > 0$, each $\theta_i \in \text{LTL}()$, O_n is '$\wedge \mathsf{G_s}$', and, for each $i < n$, O_i is either 'U' or 'U_+' or '$\wedge \mathsf{X}$'. Further, let $\mathcal{B} \subseteq \text{LTL}()$ be a finite set. An α-formula is defined as

$$\alpha(\delta,\mathcal{B}) = \big(\theta_1 O_1 (\theta_2 O_2 \ldots (\theta_n O_n \theta_{n+1}) \ldots)\big) \wedge \bigwedge_{\psi \in \mathcal{B}} \mathsf{G_s}\mathsf{F_s}\psi$$

Hence, a word u satisfies $\alpha(\delta,\mathcal{B})$ iff u can be written as $u_1.u_2.\cdots.u_{n+1}$, where

- each u_i consists only of actions satisfying θ_i and
  - $|u_i| \geq 0$ if $i = n+1$ or O_i is 'U',
  - $|u_i| > 0$ if O_i is 'U_+',
  - $|u_i| = 1$ if O_i is '$\wedge \mathsf{X}$' or '$\wedge \mathsf{G_s}$',
- and u_{n+1} satisfies $\mathsf{G_s}\mathsf{F_s}\psi$ for every $\psi \in \mathcal{B}$.

Proof of the following lemma is a simple exercise.

Lemma 1. *A conjunction of α-formulae can be effectively converted into an equivalent disjunction of α-formulae.*

Theorem 2. *Every LTL$(\mathsf{F_s},\mathsf{G_s})$ formula can be translated into an equivalent disjunction of α-formulae.*

Proof (Sketch). Given an LTL($\mathsf{F}_s, \mathsf{G}_s$) formula φ, we construct a finite set A_φ of α-formulae such that φ is equivalent to disjunction of formulae in A_φ. The proof proceeds by induction on the length of φ. The base case shows that the theorem holds for all formulae of LTL(). The inductive step is done by a detailed analysis of the structure of φ (it distinguishes 19 cases). □

4 Model Checking of wPRS Against Negated $\mathcal{A}$

This section is devoted to decidability of the model checking problem for wPRS and negated formulae of the $\mathcal{A}$ fragment. In fact, we prove decidability of the dual problem, i.e. whether a given wPRS system has a run satisfying a given formula of $\mathcal{A}$. Finite and infinite runs are treated separately.

Theorem 3. *The problem whether a given wPRS system has a finite run satisfying a given α-formula is decidable.*

Proof (Sketch). The problem is reduced to the reachability Hennessy–Milner property problem, which is decidable for wPRS [KŘS05]. □

The problem for infinite runs is more complicated. In order to solve it, we introduce more terminology and notation. First we define β-*formulae* and regular languages called γ-*languages*. Let $w = a_1 O_1 a_2 O_2 \dots a_n O_n$, where $n \geq 0$, $a_1, \dots, a_n \in Act$ are pairwise distinct actions and each O_i is either 'U_+' or '$\wedge\mathsf{X}$'. Further, let $B \subseteq Act \setminus \{a_1, \dots, a_n\}$ be a nonempty finite set of actions and $C \subseteq B$. A β-formula $\beta(w,B,C)$ and γ-language $\gamma(w,C)$ are defined as

$$\beta(w,B,C) = \big(a_1 O_1 (a_2 O_2 \dots (a_n O_n \mathsf{G} \bigvee_{b \in B} b) \dots)\big) \wedge \bigwedge_{b \in C} \mathsf{GF}b \wedge \bigwedge_{b \in B \setminus C} (\mathsf{F}b \wedge \neg \mathsf{GF}b)$$

$$\gamma(w,C) = a_1^{o_1}.a_2^{o_2}.\cdots.a_n^{o_n}.L,$$

$$\text{where } o_i = \begin{cases} + & \text{if } O_i = \mathsf{U}_+ \\ 1 & \text{if } O_i = \wedge\mathsf{X} \end{cases} \quad \text{and } L = \begin{cases} \{\varepsilon\} & \text{if } C = \emptyset \\ \bigcap_{b \in C} C^*.b.C^* & \text{otherwise} \end{cases}$$

Roughly speaking, a β-formula is a more restrictive version of an α-formula and in context of β-formulae we consider infinite words only. Contrary to δ of an α-formula, w of a β-formula employs actions rather than LTL() formulae. While a tail of an infinite word satisfying an α-formula is specified by θ_{n+1}, in the definition of β-formulae we use a set B containing exactly all the actions of the tail and its subset C of exactly all actions occurring infinitely many times in the tail.

Note that an infinite word satisfies a formula $\beta(w,B,C)$ if and only if it can be divided into a prefix $u \in \gamma(w,B)$ and a suffix $v \in C^\omega$ such that v contains infinitely many occurrences of every $c \in C$.

Let w,B,C be defined as above. We say that a finite derivation σ over a word u *satisfies* $\gamma(w,C)$ if and only if $u \in \gamma(w,C)$. We write $(w',B') \sqsubseteq (w,B)$ whenever $B' \subseteq B$ and $w' = a_{i_1} O_{i_1} a_{i_2} O_{i_2} \dots a_{i_k} O_{i_k}$ for some $1 \leq i_1 < i_2 < \dots < i_k \leq n$. Moreover, we write $(w',B',C') \sqsubseteq (w,B,C)$ whenever $(w',B') \sqsubseteq (w,B)$, B' is nonempty, and $C' \subseteq C \cap B'$.

Remark 4. If u is an infinite word satisfying $\beta(w,B,C)$ and v is an infinite *subword* of u (i.e. it arises from u by omitting some letters), then there is exactly one triple $(w',B',C') \sqsubseteq (w,B,C)$ such that $v \models \beta(w',B',C')$. Further, for each finite subword v of u, there is exactly one pair (w',B') such that $(w',B') \sqsubseteq (w,B)$ and $v \in \gamma(w',B')$.

Given a PRS in normal form, by $tri(\Delta)$, $par(\Delta)$, and $seq(\Delta)$ we denote the system Δ restricted to trivial, parallel, and sequential rules, respectively. A derivation in $tri(\Delta)$ is called a *trivial* derivation in Δ. In the following we write simply tri, par, seq as Δ is always clearly determined by the context.

Definition 5. *Let Δ be a PRS in normal form and $\beta(w,B,C)$ be a β-formula. The PRS Δ is in* flat (w,B,C)-form *if and only if for each $X,Y \in Const(\Delta)$, each $(w',B',C') \sqsubseteq (w,B,C)$, and each $B'' \subseteq B$, the following conditions hold:*

1. *If there is a finite derivation $X \xrightarrow{u} Y$ satisfying $\gamma(w',B'')$, then there is also a finite derivation $X \xrightarrow{v}_{tri} Y$ satisfying $\gamma(w',B'')$.*
2. *If there is a term t and a finite derivation $X \xrightarrow{u} t$ satisfying $\gamma(w',B'')$, then there is also a constant Z and a finite derivation $X \xrightarrow{v}_{tri} Z$ satisfying $\gamma(w',B'')$.*
3. *If $w' = \varepsilon$ and there is an infinite derivation $X \xrightarrow{u}$ satisfying $\beta(w',B',C')$, then there is also an infinite derivation $X \xrightarrow{v}_{tri}$ satisfying $\beta(w',B',C')$.*
4. *If there is an infinite derivation $X \xrightarrow{u}_{par}$ satisfying $\beta(w',B',C')$, then there is also an infinite derivation $X \xrightarrow{v}_{tri}$ satisfying $\beta(w',B',C')$;*
5. *If there is an infinite derivation $X \xrightarrow{u}_{seq}$ satisfying $\beta(w',B',C')$, then there is also an infinite derivation $X \xrightarrow{v}_{tri}$ satisfying $\beta(w',B',C')$.*

Intuitively, the system is in flat (w,B,C)-form if for every derivation of one of the listed types there is an "equivalent" trivial derivation. All conditions of the definition can be checked due to the following lemma, [Boz05], and decidability of LTL model checking for PDA and PN. Lemma 7 says that every PRS in normal form can be transformed into an "equivalent" flat system. Finally, the Lemma 10 says that if a PRS system in flat (w,B,C)-form has an infinite derivation satisfying $\beta(w,B,C)$, then it has also a trivial infinite derivation satisfying $\beta(w,B,C)$. Note that it is easy to check whether such a trivial derivation exists.

Lemma 6. *Given a γ-language $\gamma(w,C)$, a PRS system Δ, and constants X,Y, the following problems are decidable:*
(i) Is there any derivation $X \xrightarrow{u} Y$ satisfying $\gamma(w,C)$?
(ii) Is there any derivation $X \xrightarrow{u} t$ such that t is a term and $u \in \gamma(w,C)$?

Proof (Sketch). Both problems can be reduced to the reachability problem for wPRS, which is known to be decidable [KŘS04a]. □

The proof of the following lemma contains the algorithmic core of this section.

Lemma 7. *Let Δ be a PRS in normal form and $\beta(w,B,C)$ be a β-formula. One can construct a PRS Δ' in flat (w,B,C)-form such that for each $(w',B',C') \sqsubseteq (w,B,C)$ and each $X \in Const(\Delta)$, Δ' is equivalent to Δ with respect to the existence of an infinite derivation starting from X and satisfying $\beta(w',B',C')$.*

Proof (Sketch). All conditions of Definition 5 can be checked for each $X, Y \in Const(\Delta)$, each $(w', B', C') \sqsubseteq (w, B, C)$, and each $B'' \subseteq B$. For Conditions 1 and 2, this follows from Lemma 6. The problem whether there is an infinite derivation $X \xrightarrow{u}$ satisfying $\beta(\varepsilon, B', C')$ is a special case of the *fairness problem*, which is decidable due to [Boz05]. Finally, Conditions 4 and 5 can be checked due to decidability of LTL model checking for PDA and PN. If there is a non-satisfied condition, we add some trivial rules forming the missing derivation. □

Definition 8 (Subderivation). *Let Δ be a PRS in normal form and σ_1 be a (finite or infinite) derivation $s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} \ldots$, where $s_1 \xrightarrow{a_1} s_2$ has the form $X \xrightarrow{a_1} Y.Z$ and, for each $i \geq 2$, if s_i is not the last state of the derivation, then it has the form $s_i = t_i.Z$ with $t_i \neq \varepsilon$. Then σ_1 is called a* subderivation *of a derivation σ if σ has a suffix σ' satisfying the following:*

1. *every transition step in σ' is of the form $s_i \| t' \xrightarrow{a_i} s_{i+1} \| t'$ or $s_i \| t' \xrightarrow{b} s_i \| t''$, where $t' \xrightarrow{b} t''$,*
2. *in σ', if we replace every step of the form $s_i \| t' \xrightarrow{a_i} s_{i+1} \| t'$ by step $s_i \xrightarrow{a_i} s_{i+1}$ and we skip every step of the form $s_i \| t' \xrightarrow{b} s_i \| t''$, we get precisely σ_1.*

Further, if σ_1 and σ are finite, the last term of σ_1 is a process constant, and σ is a prefix of a derivation σ', then σ_1 is also a subderivation *of σ'.*

Remark 9. Let Δ be a PRS in normal form and σ be a derivation of Δ having a suffix σ' of the form $\sigma' = X \| t \xrightarrow{a} (Y.Z) \| t \xrightarrow{u}$. Then, there is a subderivation of σ whose first transition step $X \xrightarrow{a} Y.Z$ corresponds to the first transition step of σ'.

Intuitively, the subderivation captures the behaviour of the subterm $Y.Z$ since its emergence until it is possibly reduced to a term without any sequential composition. Due to the normal form of Δ, the subterm $Y.Z$ behaves undependently on the rest of the term (as long as it contains a sequential composition).

Lemma 10. *Let Δ be a PRS in flat (w, B, C)-form. Then, the following condition holds for each $X \in Const(\Delta)$ and each $(w', B', C') \sqsubseteq (w, B, C)$:*
If there is an infinite derivation $X \xrightarrow{u}$ satisfying $\beta(w', B', C')$, then there is also an infinite derivation $X \xrightarrow{v}_{tri}$ satisfying $\beta(w', B', C')$.

Proof (Sketch). Given an infinite derivation σ satisfying a formula $\beta(\sigma) = \beta(w', B', C')$ where $(w', B', C') \sqsubseteq (w, B, C)$, by *trivial equivalent* of σ we mean an infinite trivial derivation starting in the same term as σ and satisfying $\beta(\sigma)$. Similarly, given a finite derivation σ satisfying some $\gamma(\sigma) = \gamma(w', B')$ where $(w', B') \sqsubseteq (w, B)$, by *trivial equivalent* of σ we mean a finite trivial derivation σ' such that σ' starts in the same term as σ, it satisfies $\gamma(\sigma)$, and if the last term of σ is a process constant, then the last term of σ' is the same process constant.

The lemma is proven by contradiction. We assume that there exist some infinite derivations violating the condition of the lemma. Let σ be one of these derivations such that the number of transition steps of σ generated by sequential non-trivial rules with actions $a \notin B$ is minimal (note that this number is always finite as we consider

derivations satisfying $\beta(w', B', C')$ for some $(w', B', C') \sqsubseteq (w, B, C)$). First, we prove that every subderivation of σ has a trivial equivalent. Then we replace all subderivations of σ by the corresponding trivial equivalents. This step is technically nontrivial because σ may have infinitely many subderivations. By the replacement we obtain an infinite derivation σ' satisfying $\beta(\sigma)$ and starting in the same process constant as σ. Moreover, σ' has no subderivations and hence it does not contain any sequential operator. Flat (w, B, C)-form of Δ (Condition 4) implies that σ' has a trivial equivalent. This is also a trivial equivalent of σ which means that σ does not violate the condition of our lemma. □

Theorem 11. *The problem whether a given PRS Δ in normal form has an infinite run satisfying a given formula $\beta(w, B, C)$ is decidable.*

Proof. Due to Lemmata 7 and 10, the problem can be reduced to the problem whether there is an infinite derivation $X \xrightarrow{v}_{tri}$ satisfying $\beta(w, B, C)$. This problem corresponds to LTL model checking of finite-state systems, which is decidable. □

The following three steps show that the previous theorem holds even for wPRS and α-formulae. The corresponding theorems and proofs can be found in [BKŘS06].

1. First we prove that the theorem holds even for α-formulae. In the proof we assign a fresh action a_θ to each subformula $\theta \in \mathrm{LTL}()$ of a given α-formula. For every such θ and every rule $t_1 \overset{a}{\hookrightarrow} t_2$ of a given PRS in normal form, if $a \models \theta$ then we add a rule $t_1 \overset{a_\theta}{\hookrightarrow} t_2$. Now we replace every θ in the α-formula by a corresponding action a_θ. The system with added rules has a run satisfying the resulting formula iff the original system has a run satisfying the original α-formula. Moreover, the resulting formula can be easily transformed into a β-formula.
2. Now we show that the system Δ does not have to be in normal form. The proof uses a modification of the standard algorithm transforming a general PRS system into an 'equivalent' PRS system in normal form [May00].
3. The last step is to move from PRS to wPRS. To remove control states from the wPRS system Δ, we replace every rule $pt_1 \overset{a}{\hookrightarrow} pt_2$ by the rule $pt_1 \overset{a_p}{\hookrightarrow} pt_2$ and every rule $pt_1 \overset{a}{\hookrightarrow} qt_2$ by the rule $pt_1 \overset{a_{p<q}}{\hookrightarrow} qt_2$. In a given α-formula, we replace every action a with $\bigvee_{p,q \in M(\Delta)} (a_p \vee a_{p<q})$. Let Δ' be the resulting PRS system and α' the resulting α-formula. We define a finite set U of α-formulae such that a run of Δ' satisfies some formula of U iff it corresponds to a correct behaviour of control unit. Hence, Δ has a run satisfying the original α-formula iff Δ' has a run satisfying α' in conjunction with one of the α-formulae of U. As conjunction of two α-formulae can be transformed into equivalent disjunction of α-formulae, we are done.

Theorem 12. *The problem whether a given wPRS system has an infinite run satisfying a given α-formula is decidable.*

As $\mathrm{LTL}(\mathsf{F_s}, \mathsf{G_s})$ is closed under negation, Theorems 2, 3, and 12 give us the following.

Corollary 13. *The model checking problem for wPRS and LTL$(\mathsf{F_s}, \mathsf{G_s})$ is decidable.*

This problem is EXPSPACE-hard due to EXPSPACE-hardness of the model checking problem for $\mathrm{LTL}(\mathsf{F}, \mathsf{G})$ for PN [Hab97]. Our decidability proof does not provide any

primitive recursive upper bound as it employs LTL model checking for PN, for which no primitive recursive upper bound is known.

5 Undecidability Results

Obviously, the model checking for wPRS and LTL(X) is decidable. Hence, to prove that the decidability boundary of Figure 2 is drawn correctly, it remains to show the following.

Theorem 14. *Model checking of PA against LTL(U) is undecidable. Model checking of PA against LTL($\overset{\infty}{\mathsf{F}}, \mathsf{X}$) is undecidable as well.*

Proof (Sketch). In both cases, the proof is done by reduction from the non-halting problem for Minsky 2-counter machine. □

In the proof of the previous theorem, the PA systems constructed there have only infinite runs. This means that model checking of infinite runs remains undecidable for PA and both LTL($\overset{\infty}{\mathsf{F}}, \mathsf{X}$) and LTL($\mathsf{U}$).

6 Conclusion

We have established the decidability border of model checking of wPRS classes and basic fragments of future LTL (see Figure 2) by showing that the model checking problem of wPRS against LTL($\mathsf{F}_s, \mathsf{G}_s$) is decidable, while the same problem for PA and LTL(U) or LTL($\mathsf{X}, \overset{\infty}{\mathsf{F}}$) is undecidable. So far, only two positive results on LTL model checking of PA (and classes subsuming PA) have been published: decidability of model checking of infinite runs for PRS and LTL fragment of fairness properties [Boz05] and decidability of the same problem for PA and *simple PLTL*$_\Box$ [BH96]. Note that the fairness fragment and the regular part of simple PLTL$_\Box$ are strictly less expressive than LTL(F, G) (also known as Lamport logic), which is again strictly less expressive than LTL($\mathsf{F}_s, \mathsf{G}_s$). We also emphasize that our positive result for LTL($\mathsf{F}_s, \mathsf{G}_s$) deals with both finite and infinite runs, and with wPRS rather than PRS or PA only.

It is also worth mentioning that our proof techniques differ from those used in [Boz05] and [BH96]. The decidability proof for LTL($\mathsf{F}_s, \mathsf{G}_s$) is based on the auxiliary result saying that model checking for wPRS and negated $\mathcal{A}$ fragment is decidable. In fact, this auxiliary result is very powerful. We conjecture that it also implies decidability of the model checking problem of wPRS and the common fragment of CTL and LTL called LTL$^{\text{det}}$ [Mai00]. Note that LTL$^{\text{det}}$ is semantically incomparable with LTL($\mathsf{F}_s, \mathsf{G}_s$).

Unfortunately, our results are insufficient to establish the decidability border for basic LTL fragments with both future and past modalities. Indeed, fragments LTL($\mathsf{F}_s, \mathsf{P}_s$) and LTL($\mathsf{F}, \mathsf{P}$), where P, P_s are past counterparts of F, F_s respectively, do not semantically coincide with any fragment of Figure 2 and decidability of the model checking problem for these two fragments and all wPRS classes subsuming PA is an open question. However, we conjecture that our technique can be adopted to answer this question positively.

Acknowledgment. Authors have been partially supported as follows: M. Křetínský by the Grant Agency of the Czech Republic, grant No. 201/06/1338, V. Řehák

by the research centre "Institute for Theoretical Computer Science (ITI)", project No. 1M0545, and J. Strejček by the Academy of Sciences of the Czech Republic, grant No. 1ET408050503. The paper has been written during J. Strejček's postdoc stay in LaBRI, Univeristé Bordeaux 1.

References

[BEM97] A. Bouajjani, J. Esparza, and O. Maler. Reachability analysis of pushdown automata: Application to model-checking. In *Proc. of CONCUR'97*, volume 1243 of *LNCS*, pages 135–150, 1997.

[BH96] A. Bouajjani and P. Habermehl. Constrained properties, semilinear systems, and petri nets. In *Proc. of CONCUR'96*, volume 1119 of *LNCS*, pages 481–497. Springer, 1996.

[BKŘS06] L. Bozzelli, M. Křetínský, V. Řehák, and J. Strejček. On Decidability of LTL Model Checking for Weakly Extended Process Rewrite Systems. Technical Report FIMU-RS-2006-05, Faculty of Informatics, Masaryk University, Brno, 2006. A full version of the paper presented at FSTTCS'06.

[Boz05] L. Bozzelli. Model checking for process rewrite systems and a class of action-based regular properties. In *Proc. of VMCAI'05*, volume 3385 of *LNCS*, pages 282–297. Springer, 2005.

[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.

[Hab97] P. Habermehl. On the complexity of the linear-time μ-calculus for Petri nets. In *Proceedings of ICATPN'97*, volume 1248 of *LNCS*, pages 102–116. Springer, 1997.

[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.

[KŘS05] M. Křetínský, V. Řehák, and J. Strejček. Reachability of Hennessy-Milner properties for weakly extended PRS. In *Proceedings of FSTTCS 2005*, volume 3821 of *LNCS*, pages 213–224. Springer, 2005.

[Lip76] R. Lipton. The reachability problem is exponential-space hard. Technical Report 62, Department of Computer Science, Yale University, 1976.

[Mai00] M. Maidl. The common fragment of CTL and LTL. In *Proc. 41th Annual Symposium on Foundations of Computer Science*, pages 643–652, 2000.

[May84] Ernst W. Mayr. An algorithm for the general Petri net reachability problem. *SIAM Journal on Computing*, 13(3):441–460, 1984.

[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.

[Min67] Marvin L. Minsky. *Computation: Finite and Infinite Machines*. Prentice-Hall, 1967.

[Pnu77] A. Pnueli. The temporal logic of programs. In *Proc. 18th IEEE Symposium on the Foundations of Computer Science*, pages 46–57, 1977.

[Str04] J. Strejček. *Linear Temporal Logic: Expressiveness and Model Checking*. PhD thesis, Faculty of Informatics, Masaryk University in Brno, 2004.