Faculty of Informatics
Masaryk University

On Extensions of Process Rewrite Systems

Vojtěch Řehák

Ph.D. Thesis Proposal

Brno, September 2004

Mojmír Křetínský
supervisor

Acknowledgment

I would like to thank Mojmír Křetínský, my supervisor, for constant willingness to discuss and help, and for his comments on a draft of this PhD thesis proposal. I would also like to thank Jan Strejček for useful collaboration.

Contents

Chapter 1

Introduction

An automatic verification of current software systems often needs to model them as infinite-state systems, i.e. systems with an evolving structure (e.g. unbounded control structures such as recursive procedure calls and/or dynamic creation of concurrent processes) and/or operating on unbounded data types: a network of mobile phones is a concurrent system with evolving structure which dynamically changes its size (and can become very large). Robustness of the network requires that underlying protocols should work for an arbitrarily large (i.e. potentially infinite) number of client processes. A JAVA applet dynamically downloads classes over the network and executes their methods, the stack of activation records should be seen as potentially infinite.

Infinite-state systems can be specified in a number of ways with their respective advantages and limitations. Petri nets, pushdown automata, and process algebras like BPA, BPP, or PA all serve to exemplify this. However a unifying view is to interpret them as labelled transition systems (LTS) with possibly infinite number of states. LTS families are often specified via a variety of rewrite systems and form hierarchies (w.r.t. bisimulation equivalence), see for example [Cau92, BCS96, Mol96, May00]. Here we employ the classes of infinite-state systems defined by term rewrite systems and called *Process Rewrite Systems* (PRS) as introduced by Mayr [May00]. PRS subsume a variety of the formalisms studied in the context of formal verification (e.g. all the models mentioned above).

A PRS is a finite set of rules $t \xrightarrow{a} t'$ where a is an action under which a subterm t can be reduced onto a subterm t'. Terms are built up from an empty process ε and a set of process constants using (associative) sequential "." and (associative and commutative) parallel "$\|$" operators. The

semantics of PRS can be defined by labelled transition systems (LTS) – labelled directed graphs whose nodes (states of the system) correspond to terms modulo properties of "." and "$\|$" and edges correspond to individual actions (computational steps) which can be performed in a given state. The relevance of various subclasses of PRS for modelling and analysing programs is shown e.g. in [Esp02], for automatic verification see e.g. surveys [BCMS01, Srb02, KJ02].

Mayr [May00] has also shown that the reachability problem (i.e. given terms t, t': is t reducible to t'?) for PRS is decidable. Most research (with some recent exceptions, e.g. [BT03, Esp02]) has been devoted to the PRS classes from the lower part of the PRS hierarchy, especially to pushdown automata (PDA), Petri nets (PN) and their respective subclasses. We mention the successes of PDA in modelling recursive programs (without process creation), PN in modelling dynamic creation of concurrent processes (without recursive calls), and CPDS (communicating pushdown systems [BET03]) modelling both features. All of these formalisms subsume a notion of a finite state unit (FSU) keeping some kind of global information which is accessible to the redices (the ready to be reduced components) of a PRS term – hence a FSU can regulate rewriting. On the other hand, using a FSU to extend the PRS rewriting mechanism is very powerful since the state-extended version of PA processes (sePA) has a full Turing-power [BEH95] – the decidability of reachability and other problems relevant for an automatic verification are lost for sePA, including all its superclasses (see Figure 2.2), and CPDS as well.

The thesis will present a hierarchy of PRS classes and their respective extensions of three types: PRS with finite constraint system (fcPRS [Str02], motivated by concurrent constraint programming, see e.g. [SR90]), state-extended PRS classes [JKM01], and our new formalism of PRS with weak finite-state unit (wPRS, introduced in [KŘS03]). In [KŘS03] we have shown that all the just mentioned extensions increase the expressive power of those PRS subclasses which do not subsume the notion of finite control. The classes in the hierarchy (depicted in Figure 2.2) are related by their expressive power with respect to (strong) bisimulation equivalence.

The notion of a weak FSU within wPRS formalism is inspired by weak automata as introduced in [MSS92], but used here as a nondeterministic (NFA) rather than alternating one. A NFA $A = (Q, \Sigma, \delta, q_0, F)$ is *weak* if its state space is partitioned into a disjoint union $Q = \bigcup Q_i$, and there is a partial order $\geq$ on the collection of the Q_i. The set Σ is the input alphabet and the transition function $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ is such that if $q \in Q_i$ and $q' \in \delta(q, a)$ then $q' \in Q_j$, where $Q_i \geq Q_j$ (this requirement on the transition

structure is also known as an *acyclicity condition*). The set F of final states satisfies that $Q_i \subseteq F$ or $Q_i \cap F = \emptyset$ for each Q_i.

As we are not interested in language equivalence, the set of final states does not play any role in our application, hence all the states of a weak NFA could belong to one class and the formalism would coincide with an arbitrary NFA. Thus we have chosen to employ a 1-weak (also known as very weak) variant of the restriction where each partition block contains exactly one state. In other words, although a weak FSU can cycle in any of its control state, each wPRS rewriting sequence can only change its state a finitely many times.

The wPRS classes refine the presented hierarchy of extended PRS formalisms and so it motivates us to focus on borders of decidability and complexity of some interesting problems. By interesting problems we mean reachability, strong and weak bisimulation equivalence, model checking problems, etc. For surveys of formal verification techniques and results see for example [Mol96, BE97, BCMS01, KJ02, Srb02].

Besides of the results on the classification of expressive power of extended PRS classes [KŘS03, KŘS04], we have shown that the reachability problem remains decidable for the very expressive class of wPRS [KŘS04]. In the context of reachability analysis one can see at least two approaches: (i) abstraction (approximate) analysis techniques on stronger 'models' such as sePA and its superclasses with undecidable reachability problem, e.g. see a recent work [BET03], and (ii) precise techniques for 'weaker' models, e.g. PRS classes with decidable reachability, e.g. [LS98] and another recent work [BT03]. In the latter one, symbolic representations of set of reachable states are built with respect to various term structural equivalences. Among others it is shown that for the PAD class and the same equivalence as in the setting presented here, when properties of sequential and parallel compositions are taken into account, one can construct nonregular representations based on counter tree automata.

In our future work we will continue in research on the refined hierarchy to obtain refined borders of decidability and complexity of some other decidability and complexity issues relevant for an automatic verification. For example, the wBPP class stands between the less expressive BPP class and the expressively stronger seBPP (the state-extended variant of BPP). On the one hand, it was proved that the strong bisimulation equivalence is in PSPACE for BPP [Jan03], on the other hand, the same problem is undecidable for the seBPP class [Mol96] (by a modification of the proof for Petri nets [Jan95b]). We conjecture that the strong bisimulation equivalence is decidable for wBPP.

By our opinion (sub)classes of wPRS are suitable for modelling some of the software systems which can be found in real-time control programs as well as in communication and cryptographic protocols. Let us mention that Hüttel and Srba [HS04] define a replicative variant of a calculus for Dolev and Yao's ping-pong protocols [DY83]. They show that the reachability problem for these protocols is decidable as it can be reduced to the reachability problem for wPRS, more precisely their replicative ping-pong protocols belong to the wPAD class.

Finally we mention another application of our decidability result exemplifying that the introduction of wPRS was well-motivated and contributes to the results on infinite-state systems. The decidability of the reachability for wPRS opens an easy way how to solve an open problem of a weak trace non-equivalence for wPRS and its subclasses.

The weak trace equivalence is defined as follows. Given a labelled transition system $(S, Act, \longrightarrow, \alpha_0)$ with a distinguished action $\tau \in Act$, we define a *weak trace set* of a state $s \in S$ (see e.g. [JEM99]) as

$$wtr(s) = \{w \in (Act \setminus \{\tau\})^* \mid s \stackrel{w}{\Longrightarrow} t \text{ for some } t \in S\},$$

where $s \stackrel{w}{\Longrightarrow} t$ means that there is some $w' \in Act^*$ such that $s \stackrel{w'}{\longrightarrow} t$ and w is equal to w' without τ actions. Two systems are *weak trace equivalent* if the weak trace sets of their initial states are the same.

Using the decidability of reachability for wPRS, it is easy to show that the weak trace set is recursive for every state of any wPRS. In other words, for every given trace w and wPRS Δ, we can decide whether w is a weak trace of Δ or not. The solution of this (i.e. weak trace set) problem for a given wPRS Δ is based on the fact that one can solve the reachability problem for a modified wPRS Δ' where the weak FSU is changed in such a way that only the traces corresponding to the weak trace w can be preformed. So far it has been known that the weak trace non-equivalence is semi-decidable for Petri nets (see e.g. [Jan95a]), pushdown processes (due to [Büc64]), and PA processes (due to [LS98]). It follows from our result that the weak trace non-equivalence is semi-decidable for wPRS. Hence, the border of the semi-decidability was moved up to the class of wPRS in the hierarchy. Let us note that the semi-decidability result is new for some classes of the "non-extended" PRS hierarchy, too; namely PAN, PAD, and PRS. As other classes of our refined hierarchy (i.e. sePA and its superclasses) have a full Turing-power, the problem is undecidable for them. Hence, the problem is solved for all classes of the refined hierarchy.

Chapter 2

State of the Art

In this chapter we briefly review some of the results concerning process rewrite systems and we also mention our results so far achieved in the context of PRS. Namely we refer to the introduction of wPRS (see Section 2.2), the results forming the expressiveness hierarchy (see Section 2.3), and the proof that the reachability problem is decidable for wPRS (see Section 2.4).

2.1 Preliminaries

A *labelled transition system (LTS)* $\mathcal{L}$ is a tuple $(S, Act, \longrightarrow, \alpha_0)$, where S is a set of *states* or *processes*, Act is a set of *atomic actions* or *labels*, $\longrightarrow \subseteq S \times Act \times S$ is a *transition relation* (written $\alpha \xrightarrow{a} \beta$ instead of $(\alpha, a, \beta) \in \longrightarrow$), $\alpha_0 \in S$ is a distinguished *initial state*.

We use the natural generalisation $\alpha \xrightarrow{\sigma} \beta$ for finite sequences of actions $\sigma \in Act^*$. The state α is *reachable* if there is $\sigma \in Act^*$ such that $\alpha_0 \xrightarrow{\sigma} \alpha$.

A binary relation R on set of states S is a *bisimulation* [Mil89] iff for each $(\alpha, \beta) \in R$ the following conditions hold:

- $\forall\, \alpha' \in S, a \in Act : \alpha \xrightarrow{a} \alpha' \Longrightarrow (\exists \beta' \in S : \beta \xrightarrow{a} \beta' \wedge (\alpha', \beta') \in R)$
- $\forall\, \beta' \in S, a \in Act : \beta \xrightarrow{a} \beta' \Longrightarrow (\exists \alpha' \in S : \alpha \xrightarrow{a} \alpha' \wedge (\alpha', \beta') \in R)$

Bisimulation equivalence (or *bisimilarity*) on a LTS is the union of all bisimulations (i.e. the largest bisimulation).

Let $Const = \{X, \ldots\}$ be a countably infinite set of *process constants*. The set $\mathcal{T}$ of *process terms* (ranged over by $t, \ldots$) is defined by the abstract syntax $t = \varepsilon \mid X \mid t_1.t_2 \mid t_1 \| t_2$, where ε is the empty term, $X \in Const$ is a

process constant (used as an atomic process), '$\|$' and '.' mean parallel and sequential compositions respectively.

The set $Const(t)$ is the set of all constants occurring in a process term t. We always work with equivalence classes of terms modulo commutativity and associativity of the parallel operator and modulo associativity of the sequential operator. We also define $\varepsilon.t = t = t.\varepsilon$ and $t\|\varepsilon = t$.

We distinguish four *classes of process terms* as: '1' stands for terms consisting of a single process constant only (e.g. $\varepsilon \notin 1$), 'S' are *sequential* terms – without parallel composition, e.g. $X.Y.Z$, 'P' are *parallel* terms – without sequential composition, e.g. $X\|Y\|Z$, and 'G' are *general* terms – with arbitrarily nested sequential and parallel compositions, like $(X.(Y\|Z))\|W$.

Definition 2.1. *Let $Act = \{a, b, \cdots\}$ be a countably infinite set of* atomic actions, *$\alpha, \beta \in \{1, S, P, G\}$ such that $\alpha \subseteq \beta$. An* (α, β)-PRS (process rewrite system) *Δ is a pair (R, t_0), where*

- *R is a finite set of* rewrite rules *of the form $t_1 \stackrel{a}{\longrightarrow} t_2$, where $t_1 \in \alpha$, $t_1 \neq \varepsilon$, $t_2 \in \beta$ are process terms and $a \in Act$ is an atomic action,*
- *$t_0 \in \beta$ is an* initial state.

Given PRS Δ we define $Const(\Delta)$ as the set of all constants occurring in the rewrite rules of Δ or in its initial state, and $Act(\Delta)$ as the set of all actions occurring in the rewrite rules of Δ. We sometimes write $(t_1 \stackrel{a}{\longrightarrow} t_2) \in \Delta$ instead of $(t_1 \stackrel{a}{\longrightarrow} t_2) \in R$.

The semantics of Δ is given by the LTS $(S, Act(\Delta), \longrightarrow, t_0)$, where $S = \{t \in \beta \mid Const(t) \subseteq Const(\Delta)\}$ is the set of states, t_0 is the initial state, and $\longrightarrow$ is the least relation satisfying the inference rules: [1]

$$\frac{(t_1 \stackrel{a}{\longrightarrow} t_2) \in \Delta}{t_1 \stackrel{a}{\longrightarrow} t_2}, \qquad \frac{t_1 \stackrel{a}{\longrightarrow} t_1'}{t_1\|t_2 \stackrel{a}{\longrightarrow} t_1'\|t_2}, \qquad \frac{t_1 \stackrel{a}{\longrightarrow} t_1'}{t_1.t_2 \stackrel{a}{\longrightarrow} t_1'.t_2}.$$

where $t_1, t_2, t_1' \in \mathcal{T}$.

If no confusion arises, we sometimes speak about a "process rewrite system" meaning a "labelled transition system generated by process rewrite system".

Obviously, it can be assumed (w.l.o.g.) the initial state t_0 of a (α, β)-PRS is a single constant as there are only finitely many terms t_i such that $t_0 \stackrel{a_i}{\longrightarrow} t_i$.

[1] Note that parallel composition is commutative and, thus, the inference rule for parallel composition also holds with t_1 and t_2 exchanged.

Figure 2.1 contains a graphical description of the hierarchy of (α, β)-PRS, simply called *PRS-hierarchy*. Some classes included in the hierarchy correspond to widely known models as finite state systems (FS), basic process algebras (BPA), basic parallel processes (BPP), process algebras (PA), pushdown processes (PDA, see [Cau92] for justification), and Petri nets (PN). The other three classes were introduced (and named) by Mayr [May00]. The hierarchy is strict w.r.t. bisimulation equivalence [May00], while it is not strict w.r.t. language equivalence. For example, both BPA and PDA define exactly the class of (ε-free) context-free languages. For more comments see the Section 2.3.

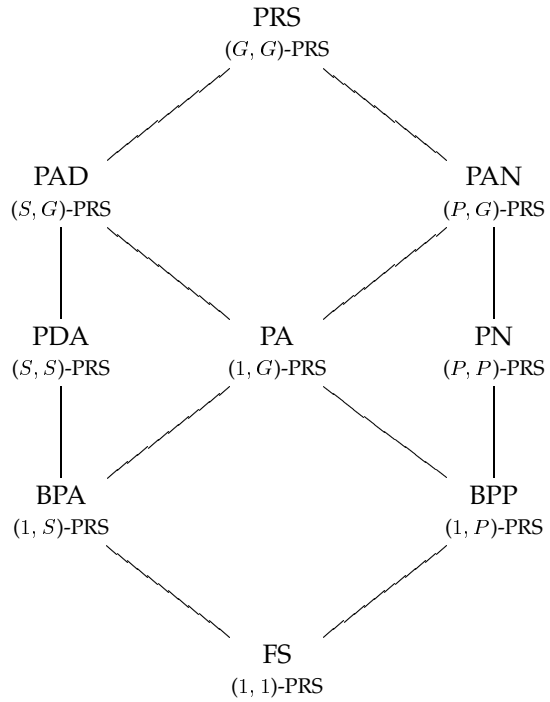

Figure 2.1: The PRS-hierarchy

2.2 Extended PRS

In this section we recall the definitions of three different extensions of process rewrite systems, namely *state-extended PRS (sePRS)* [JKM01], *PRS with a finite constraint system (fcPRS)* [Str02], and *PRS with a weak finite-state unit*

(wPRS) [KŘS03]. In all cases, the PRS formalism is extended with a finite state unit of some kind.

sePRS State-extended PRS corresponds to PRS extended with a finite state unit without any other restrictions. The well-known example of this extension is the state-extended BPA class (also known as pushdown processes).

wPRS The notion of weakness employed in the wPRS formalism corresponds to that of weak automaton [MSS92] in automata theory. The behaviour of a weak state unit is acyclic, i.e. states of state unit are ordered and non-increasing during every sequence of actions. As the state unit is finite, its state can be changed only finitely many times during every sequence of actions.

fcPRS The extension of PRS with finite constraint systems is motivated by *concurrent constraint programming (CCP)* (see e.g. [SR90]). In CCP the processes work with a shared *store* (seen as a constraint on values that variables can represent) via two operations, *tell* and *ask*. The *tell* adds a constraint to the store provided the store remains *consistent*. The *ask* is a test on the store – it can be executed only if the current store implies a specified constraint.

Formally, values of a store form a bounded lattice (called a *constraint system*) with the lub operation $\wedge$ (least upper bound), the least element *tt*, and the greatest element *ff*. The execution of *tell(n)* changes the value of the store from o to $o \wedge n$ (provided $o \wedge n \neq ff$ – consistency check). The *ask(m)* can be executed if the current value of the store o is greater than m.

The state unit of fcPRS has the same properties as the store in CCP. We add two constraints (m, n) to each rewrite rule. The application of a rule corresponds to the concurrent execution of *ask(m)*, *tell(n)*, and rewriting:

- a rule can be applied only if the actual store o satisfies $m \leq o$ and $o \wedge n \neq ff$,
- the application of the rule rewrites the process term and changes the store to $o \wedge n$.

We first define the common syntax of the aforementioned extended PRS and then we specify the individual restrictions on state units.

Definition 2.2. *Let* $Act = \{a, b, \cdots\}$ *be a countably infinite set of* atomic actions, $\alpha, \beta \in \{1, S, P, G\}$ *such that* $\alpha \subseteq \beta$. *An* extended (α, β)-PRS Δ *is a tuple* $(M, \leq, R, m_0, t_0)$, *where*

- M *is a finite set of* states *of the* state unit,
- $\leq$ *is a binary relation over* M,
- R *is a finite set of* rewrite rules *of the form* $(m, t_1) \xrightarrow{a} (n, t_2)$, *where* $t_1 \in \alpha$, $t_1 \neq \varepsilon$, $t_2 \in \beta$, $m, n \in M$, *and* $a \in Act$,
- *Pair* $(m_0, t_0) \in M \times \beta$ *forms a distinguished* initial state *of the system.*

The specific type of an extended (α, β)-PRS is given by further requirements on $\leq$. An extended (α, β)-PRS is

- (α, β)-*sePRS* without any requirements on $\leq$.[2]
- (α, β)-*wPRS* iff $(M, \leq)$ is a partially ordered set.
- (α, β)-*fcPRS* iff $(M, \leq)$ is a bounded lattice. The lub operation (least upper bound) is denoted by $\wedge$, the least and the greatest elements are denoted by *tt* and *ff*, respectively. We also assume that $m_0 \neq ff$.

To shorten our notation we prefer mt over (m, t). As in the PRS case, instead of $(mt_1 \xrightarrow{a} nt_2) \in R$ where $\Delta = (M, \leq, R, m_0, t_0)$, we usually write $(mt_1 \xrightarrow{a} nt_2) \in \Delta$. The meaning of $Const(\Delta)$ (process constants used in rewrite rules or in t_0) and $Act(\Delta)$ (actions occurring in rewrite rules) for a given extended PRS Δ is also the same as in the PRS case.

The semantics of an extended (α, β)-PRS system Δ is given by the corresponding labelled transition system $(S, Act(\Delta), \longrightarrow, m_0 t_0)$, where[3]

$$S = M \times \{t \in \beta \mid Const(t) \subseteq Const(\Delta)\}$$

and the relation $\longrightarrow$ is defined as the least relation satisfying the inference rules corresponding to the application of rewrite rules (and dependent on the concrete formalism):

$$\text{sePRS} \qquad \frac{(mt_1 \xrightarrow{a} nt_2) \in \Delta}{mt_1 \xrightarrow{a} nt_2}$$

$$\text{wPRS} \qquad \frac{(mt_1 \xrightarrow{a} nt_2) \in \Delta}{mt_1 \xrightarrow{a} nt_2} \text{ if } n \leq m$$

$$\text{fcPRS} \qquad \frac{(mt_1 \xrightarrow{a} nt_2) \in \Delta}{ot_1 \xrightarrow{a} (o \wedge n)t_2} \text{ if } m \leq o \text{ and } o \wedge n \neq ff$$

[2] In this case, the relation $\leq$ can be omitted from the definition.

[3] If Δ is an fcPRS, we eliminate the states with *ff* from S as they are unreachable.

and two common inference rules

$$\frac{mt_1 \xrightarrow{a} nt_1'}{m(t_1 \| t_2) \xrightarrow{a} n(t_1' \| t_2)}, \qquad \frac{mt_1 \xrightarrow{a} nt_1'}{m(t_1.t_2) \xrightarrow{a} n(t_1'.t_2)},$$

where $t_1, t_2, t_1' \in \mathcal{T}$ and $m, n, o \in M$.

Instead of $(1, S)$-sePRS, $(1, S)$-wPRS, $(1, S)$-fcPRS, ... we use a more natural notation seBPA, wBPA, fcBPA, etc. The class seBPP is also known as *multiset automata (MSA)* or *parallel pushdown automata (PPDA)*, see [Mol96].

2.3 Expressiveness

Figure 2.2 describes the hierarchy of PRS classes and their extended counterparts with respect to bisimulation equivalence. If any process in class X can be also defined (up to bisimilarity) in class Y we write $X \subseteq Y$. If additionally $Y \not\subseteq X$ holds, we write $X \subsetneq Y$ and say X is less expressive than Y. This is depicted by the line(s) connecting X and Y with Y placed higher than X in Figure 2.2. The dotted lines represent the facts $X \subseteq Y$, where we conjecture that $X \subsetneq Y$ hold.

Some observations (even up to isomorphism) are immediate, for example (i) collapses of the classes FS, PDA and PN with their extended analogues, (ii) if X $\subseteq$ Y then {se,w,fc}X is less expressible then the corresponding extension of Y, and (iii) (α, β)-PRS $\subseteq$ (α, β)-fcPRS $\subseteq$ (α, β)-wPRS $\subseteq$ (α, β)-sePRS for every (α, β)-PRS class.

The strictness ('$\subsetneq$') between the PRS-hierarchy classes has been proved by Mayr [May00], the strictness between the corresponding classes of PRS and fcPRS has been proved in [Str02], and the strictness relating fcPRSs and wPRSs is shown in [KŘS03] (by proving there are the following system: a PDA which is not bisimilar to any wPAN, a MSA which is not bisimilar to any wPAD, a wBPP which is not bisimilar to any fcPAD and to any PAD as well, and wBPA which is not bisimilar to any fcPAN). Note the strictness relations wX $\subsetneq$ seX hold for all X = PA, PAD, PAN, PRS due to our reachability result for wPRS given in [KŘS04] and due to the full Turing-power of sePA [BEH95].

These proofs together with Moller's result establishing that MSA $\subsetneq$ PN [Mol98] complete the justification of Figure 2.2 – with one exception, namely the relation between the PN and sePA classes. Looking at two lines leaving sePA down to the left and down to the right, we note the "left-part collapse" of (S, S)-PRS and PDA proved by Caucal [Cau92] (up to

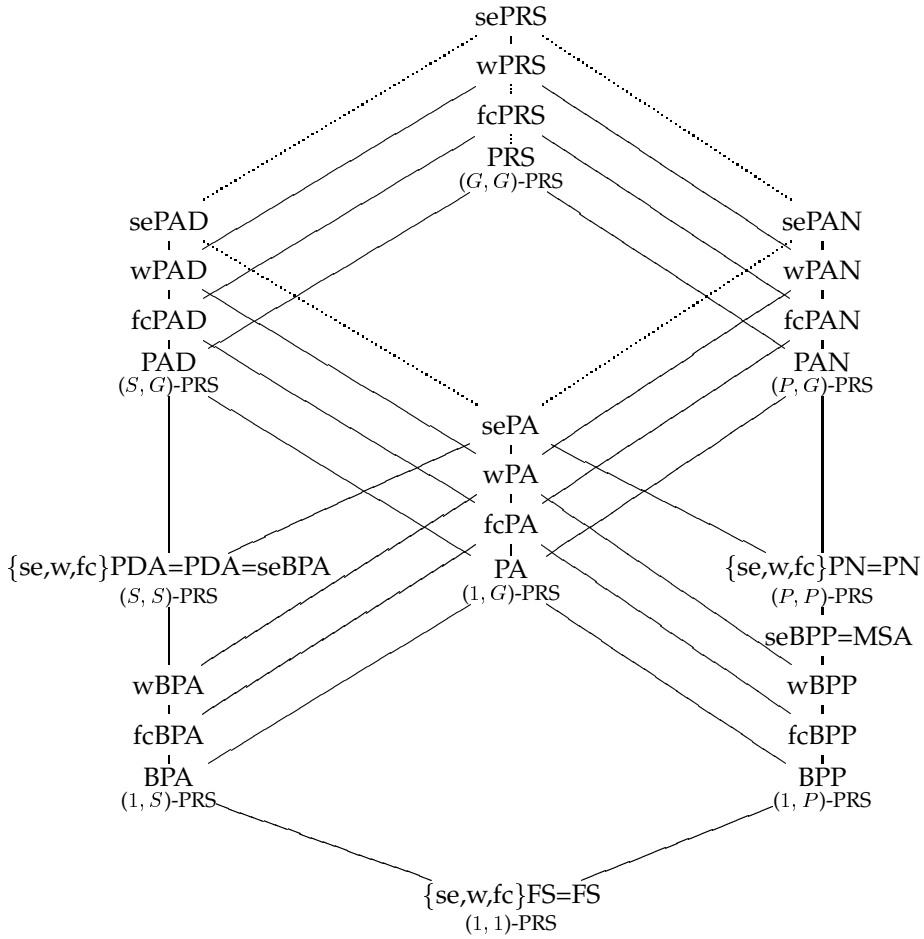

Figure 2.2: The hierarchy of classes defined by extended process rewrite systems with respect to the strong bisimulation equivalence.

isomorphism). The right-part counterpart is slightly different due to the previously mentioned result that MSA $\subsetneq$ PN. In [KŘS04] we proved that PN $\subsetneq$ sePA (in fact it suffices to demonstrate PN $\subseteq$ sePA as the strictness is obvious).

2.4 Reachability Problem

By the reachability problem we mean to decide for a wPRS Δ with an initial state rt_0 and a given state st, whether the state st is reachable from the initial state rt_0 or not (st is reachable from rt_0 if a sequence of actions σ such that $rt_0 \stackrel{\sigma}{\longrightarrow} st$ exists).

It was proved by Mayr [May00] that the reachability problem is decidable for PRS. In [MR98] Mayr and Rusinowitch present a simpler and more readable proof in the framework of ground AC rewrite systems. Bouajjani and Touili [BT03] develop automata techniques allowing to build finite representations of the forward and backward sets of reachable configurations of PRS modulo various term structural equivalences (corresponding to properties of the sequential operator and the parallel operator).

As already mentioned, we have shown the reachability problem for wPRS remains decidable. Our proof exhibits a similar structure to the proof of decidability of the reachability problem for PRS [May00]; first we reduce the general problem to the reachability problem for wPRS in so-called normal form (i.e. PRS with rules containing at most one occurrence of a sequential or parallel operator), and then we solve this subproblem using the fact that the reachability problems for both PN and PDA are decidable [May81, Büc64]. The latter part of Mayr's proof for PRS transforms the PRS Δ in normal form into the PRS Δ' in so-called *transitive normal form* satisfying $(X \longrightarrow Y) \in \Delta'$ whenever $X \succ^{\Delta} Y$. This step employs the local effect of rewriting under sequential rules in a parallel environment and vice versa. Intuitively, whenever there is a rewriting sequence

$$X \| Y \longrightarrow (X_1.X_2) \| Y \longrightarrow (X_1.X_2) \| Z \longrightarrow X_2 \| Z$$

in a PRS in normal form, then the rewriting of each parallel component is independent in the sense that there are also rewriting sequences $X \longrightarrow X_1.X_2 \longrightarrow X_2$ and $Y \longrightarrow Z$. This does not hold for wPRS in normal form as the rewriting in one parallel component can influence the rewriting in other parallel components via a weak state unit. To get this independence back we introduced the concept of *passive steps* emulating the changes of a weak state produced by the environment. For more details see [KŘS04].

To sum up, we proved that the reachability problem for wPRS is decidable. As "stronger" classes (i.e. sePA and its superclasses) of the hierarchy are Turing powerful (thus the problem is undecidable for them), the problem is solved for all the classes in the refined hierarchy.

Consequences and applications of our result on the reachability for wPRS are discussed in the introduction and Section 2.3.

2.5 History and Related Results

The first approach to process rewrite system relating various classes of PRS can be mentioned as Caucal works [CM90, Cau92]. [CM90] shows that BPA are strictly less expressive than PDA with respect to strong bisimulation. [Cau92] studies so called "prefix rewriting system" (which are also known as sequential PRS [May00]) and implicitly contains their hierarchy. To mention some early works on parallel PRS we note the results [Huy83, Huy85] for "commutative grammars" covering both BPP and PN. Later the systems were studied as generators of transition systems, rather than language generators. Let us refer to the work [CHM93] describing BPP as a subclass of CCS process algebra. In [Hir94, Esp97] the BPP class was also studied as a subclass of Petri nets.

First Chomsky-like hierarchy classification of rewrite systems can be found in [Cau92]. Moller presented the hierarchy enlarged by the parallel dimension in [Mol96]. The general process rewrite hierarchy with all combinations of the sequential operator and the parallel operator was introduced by Mayr in his PhD thesis [May98] and later published in [May00].

As the range of rewrite systems is very large, results are shattered in plenty of papers. In spite of this, there are some papers that survey the area. The first overview was done by Moller in his paper "Infinite results" [Mol96]. The paper was followed by Burkart and Esparza "More Infinite Results" paper [BE97] focused on the model checking problem. As later survey papers see "Verification on Infinite Structures" in handbook of process algebra [BCMS01], overview paper "Equivalence-Checking with Infinite-State Systems: Techniques and Results" [KJ02] written by Kučera and Jančar, and "Roadmap of Infinite results" [Srb02] built by Srba.

Besides of the summarisations containing some outdated information, the "Roadmap of Infinite results" is still keep up to date (see electronic version at `http://www.brics.dk/~srba/roadmap/` or new published version in [Srb04]) and surveys all known results for PRS hierarchy concerning decidability and complexity of strong and weak bisimilarity, strong and weak bisimilarity with finite-state systems, and strong and weak regularity.

Chapter 3

Aim of the Thesis

3.1 Objectives

The thesis will gather results related to my work on various extensions of process rewrite systems especially those extended with weak finite-state unit (see Section 2.2). It will include the just mentioned results (see Section 2.3 and Section 2.4); namely the results forming the expressiveness hierarchy with respect to the strong bisimulation [KŘS03, KŘS04] and the proof that the reachability problem is decidable for wPRS [KŘS04]. The thesis will be also composed of new topics I will focus on in my future work. For more detailed description of the future work see Section 3.3.

3.2 Progression Schedule

I have already passed a doctoral (PhD) examination in April 2003. My plan of progression (besides of other publication activity) is as follows.

- Defence of this thesis proposal - January 2005
- Final version of thesis - January 2006
- Defence of the thesis - May 2006

3.3 Expected Results and Outputs

I will study the expressiveness and the other properties of the extended process rewrite systems. I would like to refine decidability and complexity borders of some interesting problems on various subclasses of wPRS and

fcPRS; such as strong and weak bisimulations, the reachability problem, the reachability property problem (i.e. question if there is a reachable state that has certain properties, see [May00]), model checking LTL or other logics, etc. As the work is focused on topics of basic research, it is very hard to predict future results. Nevertheless, in the near future I will focus on the reachability property problem for wPRS and the (un)decidability question of strong bisimulation. I hope for decidability of strong bisimulation for the wBPP class using the proof technique published in [Jan03].

I expect at least two other reviewed publications related to the topic of the thesis.

Chapter 4

Current Results

This chapter consists of three sections. The first section summarises my previous research on binary decision diagrams, the basic data structure for symbolic model checkers, while the second contains the results related to the topic of the intended PhD thesis. The last section covers other publications related to my work on hardware design verification.

4.1 Symbolic Model Checking

My master thesis [1] studies improvements of the symbolic model checking algorithm. I investigated possible substitutions of the currently used OBDD data structure. Two data structures, $\oplus$-OBDDs and BEDs, are tested. The benefit of the thesis consists of accumulating the relevant theoretical knowledge, introduction of merged $\oplus$-OBDD, practical implementation of $\oplus$-OBDDs into the NuSMV model checker, and interpretation of the acquired facts. The second publication [2] is a conference paper which summarise some of the results achieved in my master thesis.

[1] V. Řehák. Randomized Symbolic Model Checking. Master's thesis, Masaryk University Brno, 38 pages, 2002.

[2] V. Řehák. $\oplus$-OBDD in Symbolic Model Checking. In M. Bielikova, editor, *Proceedings of SOFSEM'02 Student Research Forum*, pages 41–46. Milovy (Czech Republic): Slovak University of Technology, 2002.

4.2 Extended Process Rewrite Systems

Publications given in this part are related to my PhD research and contains partial results that will be included in my PhD thesis. The first paper [1] is an introduction of process rewrite systems with weak finite-state unit presented at Infinity'03 workshop (also it should appear in post-workshop proceedings within a volume of ENTCS - accepted for publication). The full version of the paper is presented as a local technical report [2]. The BRICS technical report [3] contains results achieved during the stay at Aarhus and forms a preliminary version of the last paper [4] presented at CONCUR 2004.

[1] M. Křetínský, V. Řehák, and J. Strejček. On Extensions of Process Rewrite Systems: Rewrite Systems with Weak Finite-State Unit. In Philippe Schnoebelen, editor, *Prelim. Proc. of the 5th International Workshop on Verification of Infinite-State Systems (INFINITY'2003)*, pages 73–86. Marseille, France: Universite de Provence, Marseille, 2003. To appear in Electronic Notes in Theoretical Computer Science. Elsevier Science Publishers.

[2] M. Křetínský, V. Řehák, and J. Strejček. Process Rewrite Systems with Weak Finite-State Unit. Technical Report FIMU-RS-2003-05, Faculty of Informatics, Masaryk University Brno, 23 pages, 2003. (full version of the INFINITY'2003 paper).

[3] M. Křetínský, V. Řehák, and J. Strejček. On the Expressive Power of Extended Process Rewrite Systems. Technical Report RS-04-07, 18 pages. Basic Research in Computer Science, Aarhus, Denmark, 2004.

[4] M. Křetínský, V. Řehák, and J. Strejček. Extended Process Rewrite Systems: Expressiveness and Reachability. In Philippa Gardner and Nobuko Yoshida, editors, *CONCUR 2004 – Concurrency Theory: 15th International Conference*, volume 3170 of *Lecture Notes in Computer Science*, pages 355–370. Elsevier Science Publishers, 2004.

4.3 Other Publications

The following publications are related to my work in the verification group focused on model checking the Liberouter design. The Liberouter project develops a dual-stack (IPv6 and IPv4) router based on the standard PC architecture.

[1] J. Barnat, T. Brázdil, P. Krčál, V. Řehák, and D. Šafránek. Model Checking in IPv6 Hardware Router Design. Technical Report 07, CESNET, 8 pages, July 2002.

[2] D. Antoš, J. Kořenek, K. Minaříková, and V. Řehák. Packet Header Matching in Combo6 IPv6 Router. Technical Report 01, CESNET, 15 pages, January 2003.

[3] D. Antoš, J. Kořenek, and V. Řehák. Vyhledávání v IPv6 směrovači implementovaném v hradlovém poli. In *EurOpen, Sborník příspěvků XXIII. konference. Strážnice: EurOpen*, pages 91–102. Strážnice, 2003.

[4] D. Antoš, V. Řehák, and J. Kořenek. Hardware Router's Lookup Machine and its Formal Verification. In *ICN'2004 Conference Proceedings*, pages 1002–1007. Gosier, Guadeloupe, French Caribbean, 2004.

[5] T. Kratochvíla, V. Řehák, and P. Šimeček. Verification of COMBO6 VHDL Design. Technical Report 17, CESNET, 17 pages, November 2003.

Bibliography

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

[BCS96] O. Burkart, D. Caucal, and B. Steffen. Bisimulation collapse and the process taxonomy. In *Proceedings of CONCUR'96*, volume 1119 of *Lecture Notes in Computer Science*, pages 247–262. Springer-Verlag, 1996.

[BE97] O. Burkart and J. Esparza. More infinite results. *Electronic Notes in Theoretical Computer Science*, 5, 1997.

[BEH95] A. Bouajjani, R. Echahed, and P. Habermehl. On the verification problem of nonregular properties for nonregular processes. In *LICS'95*, pages 123–133. IEEE Computer Society Press, 1995.

[BET03] A. Bouajjani, J. Esparza, and T. Touili. A generic approach to the static analysis of concurrent programs with procedures. *International Journal on Foundations of Computer Science*, 14(4):551–582, 2003.

[BT03] A. Bouajjani and T. Touili. Reachability Analysis of Process Rewrite Systems. In *Proceedings of FST&TCS-2003*, volume 2914 of *Lecture Notes in Computer Science*, pages 74–87. Springer-Verlag, 2003.

[Büc64] J. R. Büchi. Regular canonical systems. *Archiv fur Mathematische Logik und Grundlagenforschung*, 6:91–111, 1964.

[Cau92] D. Caucal. On the regular structure of prefix rewriting. *Theoretical Computer Science*, 106:61–86, 1992.

[CHM93] S. Christensen, Y. Hirshfeld, and F. Moller. Bisimulation is decidable for all basic parallel processes. In *Proceedings of CONCUR'93*, volume 715 of *Lecture Notes in Computer Science*, pages 143–157. Springer-Verlag, 1993.

[CM90] D. Caucal and R. Monfort. On the transition graphs of automata and grammars. In *Graph-Theoretic Concepts in Computer Science*, volume 484 of *Lecture Notes in Computer Science*, pages 311–337, Berlin, 1990. Springer-Verlag.

[DY83] D. Dolev and A. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, 1983.

[Esp97] J. Esparza. Petri nets, commutative context-free grammars, and basic parallel processes. *Fundamenta Informaticae*, 31(1):13–25, July 1997.

[Esp02] J. Esparza. Grammars as processes. In J. karhuaki W. Brauer, H. Ehrig and A. Salomaa, editors, *Formal and Natural Computing*, volume 2300 of *Lecture Notes in Computer Science*, pages 277–297. Springer-Verlag, 2002.

[Hir94] J. Hirshfeld. Petri nets and the equivalence problem. In *Computer Science Logic, 7th Workshop, CSL '93*, volume 832 of *Lecture Notes in Computer Science*, pages 165–174. Springer-Verlag, 1994.

[HS04] H. Hüttel and J. Srba. Recursion vs. replication in simple cryptographic protocols. Submitted for publication, 2004.

[Huy83] D. T. Huynh. Commutative grammars: The complexity of uniform word problems. *Information and Control*, 57(1):21–39, April 1983.

[Huy85] D. T. Huynh. The complexity of equivalence problems for commutative grammars. *Information and Control*, 66(1/2):103–121, July/August 1985.

[Jan95a] P. Jančar. High Undecidability of Weak Bisimilarity for Petri Nets. In *Proceedings of TAPSOFT*, volume 915 of *Lecture Notes in Computer Science*, pages 349–363. Springer-Verlag, 1995.

[Jan95b] P. Jančar. Undecidability of bisimilarity for Petri nets and some related problems. *Theoretical Computer Science*, 148(2):281–301, 1995.

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

[JEM99] P. Jančar, J. Esparza, and F. Moller. Petri nets and regular behaviours. *Journal of Computer and System Sciences*, 59(3):476–503, 1999.

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

[KJ02] A. Kučera and P. Jančar. Equivalence-checking with infinite-state systems: Techniques and results. In *Proceedings of SOFSEM'2002*, volume 2540 of *Lecture Notes in Computer Science*, pages 41–73. Springer-Verlag, 2002.

[KŘS03] M. Křetínský, V. Řehák, and J. Strejček. Process Rewrite Systems with Weak Finite-State Unit. Technical Report FIMU-RS-2003-05, Masaryk University Brno, 2003. 23 pp. To appear in Electronic Notes in Theoretical Computer Science as Proceedings of INFINITY 03.

[KŘS04] M. Křetínský, V. Řehák, and J. Strejček. Extended Process Rewrite Systems: Expressiveness and Reachability. In Philippa Gardner and Nobuko Yoshida, editors, *CONCUR 2004 – Concurrency Theory: 15th International Conference*, volume 3170 of *Lecture Notes in Computer Science*, pages 355–370. Elsevier Science Publishers, 2004.

[LS98] D. Lugiez and Ph. Schnoebelen. The regular viewpoint on PA-processes. In *Proceedings of CONCUR'98*, volume 1466 of *Lecture Notes in Computer Science*, pages 50–66, 1998.

[May81] E. W. Mayr. An algorithm for the general Petri net reachability problem. In *Proceedings of 13th Symposium on Theory of Computing*, pages 238–246. ACM Press, 1981.

[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 *Proceedings of CONCUR'96*, volume 1119 of *Lecture Notes in Computer Science*, pages 195–216. Springer-Verlag, 1996.

[Mol98] F. Moller. A Taxonomy of Infinite State Processes, MFCS'98 Workshop on Concurrency. *Electronic Notes in Theoretical Computer Science*, 18, 1998.

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

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

[SR90] V. A. Saraswat and M. Rinard. Concurrent constraint programming. In *Proceedings of 17th POPL*, pages 232–245. ACM Press, 1990.

[Srb02] J. Srba. Roadmap of infinite results. *Bulletin of the European Association for Theoretical Computer Science*, (78):163–175, 2002. `http://www.brics.dk/~srba/roadmap/`.

[Srb04] J. Srba. *Current Trends In Theoretical Computer Science, The Challenge of the New Century*, volume 2: Formal Models and Semantics, chapter Roadmap of Infinite results, pages 337–350. World Scientific Publishing Co., 2004.

[Str02] J. Strejček. Rewrite systems with constraints, EXPRESS'01. *Electronic Notes in Theoretical Computer Science*, 52, 2002.

Appendix A

Summary / Souhrn

The intended thesis will be focused on properties of Process Rewrite Systems (PRS). Namely, it will introduce an extension of PRS, so called Process Rewrite Systems with Weak Finite-State Unit (wPRS). The thesis will compare the expressiveness of wPRS with original PRS classes and their known extensions. In addition, it will study decidability and complexity of problems related to model checking and other formal verification procedures such as weak and strong bisimulation, the reachability problem, etc. The aim of the thesis is to extend expressive power of known modelling facilities while preserving decidability and maintaining complexity of problems in reasonable bounds.

Zamýšlená disertační práce se zaměří na vlastnosti procesových přepisovacích systémů (PRS). Práce představí nové rozšíření hierarchie procesových přepisovacích systémů o konečně stavovou jednotkou (wPRS) a porovná vyjadřovací sílu těchto tříd s dosud známými rozšířeními. Dále bude práce studovat hranice rozhodnutelnosti a složitosti zajímavých problémů vztahujících se k ověřování vlastností modelů, či jiným metodám formální verifikace. Ze zajímavých problémů jmenujme například problém dosažitelnosti, či problémy rozhodování silné a slabé bisimulace. Cílem práce je rozšířit vyjadřovací sílu známých modelů o přirozeně motivované možnosti, které však v rozumné míře zachovají rozhodnutelnost zmiňovaných problémů a složitostní odhady algoritmů řešících tyto problémy.