Combining Petri Nets and PA-Processes

Richard Mayr

Institut für Informatik, Technische Universität München,
Arcisstr. 21, D-80290 München, Germany;
e-mail: mayrri@informatik.tu-muenchen.de
fax: +49 (89) 289-28207/28483

Abstract. Petri nets and PA-processes are incomparable models of infinite state concurrent systems. We define a new model PAN (= PA + PN) that subsumes both of them and is thus strictly more expressive than Petri nets. It extends Petri nets with the possibility to call subroutines. We show that the reachability problem is still decidable for PAN. It is even decidable, if there is a reachable state that satisfies certain properties that can be encoded in a simple logic.

Keywords: Petri nets, PA-processes, process algebra, reachability problem

1 Introduction

PA (Process algebra) [BEH95, Kuc96] is the name that has become common use to denote the algebra with a sequential and parallel operator (without communication), plus recursion. PA-processes are a superset of both Basic Parallel Processes (BPP) [Chr93] and context-free processes (BPA). PA-processes and Petri nets are incomparable models of concurrent systems, meaning that neither model is more expressive than the other one.

Here we introduce PAN (Process Algebra Nets), which is the smallest extension of Petri nets that contains PA-processes. It is strictly more expressive than Petri nets and can thus be used to model systems that exceed the bounds of the expressiveness of Petri nets. The main extension over Petri nets is that it is possible to call subroutines. Thus it is possible to model recursion and tree-like structures. As PAN is a very expressive model, model checking with most common temporal logics (except Hennesy-Milner logic) is undecidable for it (see section 2). However, we show that the reachability problem is still decidable for PAN. This generalizes the results on decidability of reachability for Petri nets [May84] and for PA-processes [May97].

In section 2 we define a general formalism for describing concurrent systems and show how Petri nets, PA-processes and PAN fit into it. In section 3 we show that the reachability problem is decidable for PAN. In section 4 we extend this result by showing that it is even decidable if there is a reachable state that satisfies certain properties that can be encoded in a simple logic. Section 5 shows an example of how systems can be modelled with PAN. The paper closes with a section that summarizes the results.

2 Preliminaries

Many classes of concurrent systems can be described by a (possibly infinite) set of process terms, representing the states and a finite set of rewrite rules describing the dynamics of the system. In this section we define this formalism and show how Petri nets and PA-processes fit into this scheme. Then we define PAN, that is the smallest extension of Petri nets that includes PA-processes.

Definition 1. Assume a countably infinite set of atomic actions $Act = \{a, b, \ldots\}$ and a countably infinite set of process variables $Var = \{X, Y, Z, \ldots\}$. The process terms that describe the states of the system have the following form:

$$\begin{array}{lll} P ::= & \epsilon & \text{empty term} \\ & X & \text{process variable} \\ & P_1.P_2 & \text{sequential composition} \\ & P_1 \| P_2 & \text{parallel composition} \end{array}$$

Convention: We always work with equivalence classes of terms modulo commutativity of parallel composition. Also we define that $\epsilon.P = P$ and $P \| \epsilon = P$.

The dynamics of the system is described by a finite set of rules Δ of the form $t_1 \stackrel{a}{\rightarrow} t_2$ where t_1 and t_2 are process terms and $a \in Act$.

This describes a labelled transition system (for short: LTS) in the following way: Let $\stackrel{a}{\rightarrow}$ be the closure of Δ under the following inference rules.

$$\frac{(t_1 \stackrel{a}{\rightarrow} t_2) \in \Delta}{t_1 \stackrel{a}{\rightarrow} t_2}$$

$$\frac{E \stackrel{a}{\rightarrow} E'}{E \| F \stackrel{a}{\rightarrow} E' \| F} \qquad \frac{F \stackrel{a}{\rightarrow} F'}{E \| F \stackrel{a}{\rightarrow} E \| F'} \qquad \frac{E \stackrel{a}{\rightarrow} E'}{E.F \stackrel{a}{\rightarrow} E'.F}$$

The relation $\stackrel{a}{\rightarrow}$ is generalized to sequences of actions in the standard way. Sequences are denoted by σ. Without restriction we can assume that the initial state of a system is described by a term consisting of a single variable.

Note that there is no operator $+$ for nondeterministic choice in the process terms, because this is encoded in the set of rules Δ !

Definition 2. Let t be a process term, then $size(t)$ denotes the size of t. Intuitively this is the number of process variables in t. The function $size$ is defined by:

$$\begin{aligned} size(\epsilon) &:= 0 \\ size(X) &:= 1 \\ size(t_1.t_2) &:= size(t_1) + size(t_2) \\ size(t_1 \| t_2) &:= size(t_1) + size(t_2) \end{aligned}$$

Note that $size$ is well-defined, because two terms that are equivalent modulo commutativity of parallel composition have the same size.

Many common models of systems fit into this scheme.

1. (a) Let all rules in Δ be of the form $X \xrightarrow{a} Y$. (Only single variables are allowed in both left and right hand sides of the rules.) Then the system is a finite state regular process.
   (b) Let all rules in Δ be of the form $X \xrightarrow{a} Y_1.Y_2.\ldots.Y_n$. (Only single variables in the left hand sides, but arbitrary sequential composition in the right hand sides.) Then we get context-free processes (BPA) [Esp97, Esp96].
   (c) If all rules are of the form $X_1.X_2.\ldots.X_i \xrightarrow{a} Y_1.Y_2.\ldots.Y_k$ (sequential composition is also allowed on the left hand sides), then the model is as expressive as pushdown processes [BS94].
2. (a) If the rules are of the form $X_1\|X_2\|\ldots\|X_i \xrightarrow{a} Y_1\|Y_2\|\ldots\|Y_k$, (arbitrary parallel composition on both sides, but no sequential composition), then we get Petri nets. Every variable corresponds to a place in the net and the number of occurrences of a variable in a term corresponds to the number of tokens in this place. Thus every rule corresponds to a transition in the net.
   (b) If all rules are of the form $X \xrightarrow{a} Y_1\|Y_2\|\ldots\|Y_n$, (only single variables on left hand sides, but arbitrary parallel composition on the right hand sides), then we get communication-free Petri nets [Esp97, Esp96]. This special class of Petri nets is equivalent to Basic Parallel Processes (BPP) [Chr93].
3. If the rules are of the form $X \xrightarrow{a} t$, where t is a general process term with arbitrary sequential and parallel composition, then we get PA-processes [May96a, BEH95, Kuc96].

PA-processes and Petri nets are incomparable models of concurrent systems. This fact is emphasized by the decidability results on the model checking problem. The simple branching time temporal logic EF (also called UB^- in [Esp97]) consists of boolean operators, the operator EX (for some successor) and EF (for some path eventually in the future). While EF is decidable for PA-processes [May96a] and PSPACE-complete for BPP [May96b], it is undecidable for Petri nets [Esp97, Esp96]. For linear time logics we have the opposite situation. While linear time μ-calculus and LTL are decidable for Petri nets [Esp97, Esp96], they are undecidable for PA-processes [BH96].

What is the simplest model of concurrent systems that subsumes both Petri nets and PA-processes ? With the formalism of Def. 1 this is straightforward:

Definition 3. PAN (Process Algebra Nets) are the processes where the rules are of the following form:

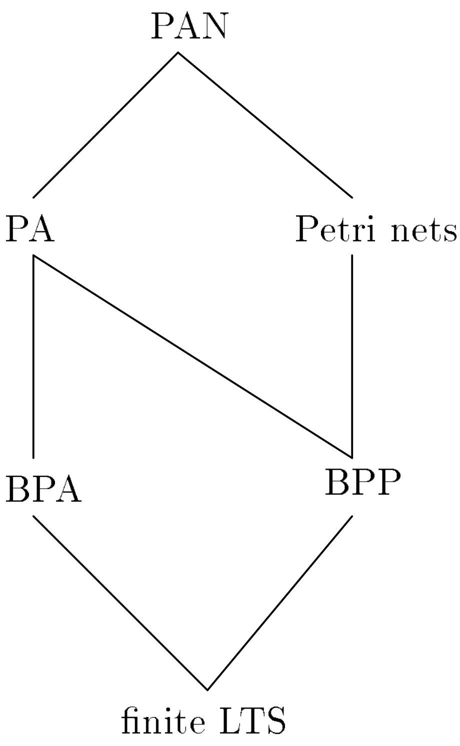
$$X_1\|X_2\|\ldots\|X_k \xrightarrow{a} t$$

Only parallel composition is allowed on the left hand sides of rules, while arbitrary sequential and parallel composition is allowed on the right hand sides.

PAN-processes are the smallest extension of Petri nets that includes PA-processes. Intuitively PAN extends Petri nets with sequential composition, which can be seen as the possibility to call subroutines.

Figure 1 shows the relationship between the various models. An arc means that the higher model is more expressive than the lower one. Models on the same level are incomparable.

Fig. 1. Relationship between BPA, BPP, Petri nets and PAN



3 The Reachability Problem

Definition 4. The reachability problem is the problem if a given state is reachable from the initial state.

Instance: A PAN Δ with initial state t_0 and a given state t.

Question: Is there a sequence σ of actions s.t. $t_0 \xrightarrow{\sigma} t$?

For Basic Parallel Processes (BPP) reachability is NP-complete [Esp97, Esp96]. Although PA-processes are more expressive than BPP, the reachability problem is still NP-complete [May97]. For Petri nets reachability is decidable and EXPSPACE-hard [May84]. Here we show that reachability is decidable for PAN.

The proof is done in two steps. First we transform the PAN into a certain normal form and show that it suffices to solve the reachability problem for this normal form. Then we prove the decidability of reachability for PAN in normal form by reducing it to the reachability problem for Petri nets.

For the reachability problem we are only interested in states, not in atomic actions or the language of the system. Therefore we ignore atomic actions and just write $t_1 \rightarrow t_2$ instead of $t_1 \xrightarrow{a} t_2$. First we will show that every PAN can be effectively transformed into an equivalent PAN that is in a certain normal form.

Definition 5. A PAN is in *normal form* if all rules in Δ are of one of the following two forms:

par-rules $X_1 \| \ldots \| X_j \rightarrow Y_1 \| \ldots \| Y_k$
seq-rules $X_1 \| \ldots \| X_j \rightarrow t_1.t_2$

For the reachability problem it suffices to regard PANs in normal form. This is because every rule of the form $X_1 \| \ldots \| X_j \rightarrow (t_1.t_2) \| (t_3.t_4)$ can be replaced by new rules $X_1 \| \ldots \| X_j \rightarrow Y_1 \| Y_2$ and $Y_1 \rightarrow t_1.t_2$ and $Y_2 \rightarrow t_3.t_4$, where Y_1 and Y_2 are new variables. In the sequel all PANs will be in normal form.

Definition 6. Let $\mathcal{T}$ be the set of all PAN-terms, $\mathcal{T}_{par}$ the set of PAN-terms without sequential composition and Δ a set of rules. Let

$$t \succ^{\Delta} t' :\iff \exists \sigma.\ t \xrightarrow{\sigma} t'$$
$$t \succ^{\Delta}_{par} t' :\iff \exists \sigma.\ t \xrightarrow{\sigma} t' \text{ and all rules in } \sigma \text{ are par-rules from } \Delta$$

If the set of rules is fixed we just write $t \succ t'$.

The following proposition follows directly from the decidability of the reachability problem for Petri nets [May84].

Proposition 7. *Let Δ be a PAN and $t, t' \in \mathcal{T}_{par}$. It is decidable if $t \succ^{\Delta}_{par} t'$.*

Lemma 8. *Let Δ be a PAN and $t \in \mathcal{T}$. It is decidable if $t \succ^{\Delta}_{par} \epsilon$.*

Proof Let *FAIL* be a new constant and $\mathcal{E}_\Delta : \mathcal{T} \cup \{FAIL\} \mapsto \{true, false\}$ a function defined by

$$\mathcal{E}_\Delta(t) := \begin{cases} false, & \text{if } FAIL \text{ occurs in } t \\ t \succ^{\Delta}_{par} \epsilon, & \text{if } t \in \mathcal{T}_{par} \\ \mathcal{E}_\Delta(t_1) \wedge \mathcal{E}_\Delta(t_2), & \text{if } t = t_1.t_2 \\ \mathcal{E}_\Delta(P(t_1 \| t_2)), & \text{if } t = t_1 \| t_2 \end{cases}$$

and $P : \mathcal{T} \mapsto \mathcal{T} \cup \{FAIL\}$ a function defined by

$$\begin{aligned} P(X) &= X \\ P(t_1 \| t_2) &= P(t_1) \ \|\ P(t_2) \\ P(t_1.t_2) &= \text{If } \mathcal{E}_\Delta(t_1) \text{ then } P(t_2) \text{ else } FAIL \end{aligned}$$

By induction on the size of t and Proposition 7 it follows that $\mathcal{E}_\Delta(t)$ is decidable. Again by induction on the size of t it can be shown that

$$\mathcal{E}_\Delta(t) \iff t \succ^{\Delta}_{par} \epsilon$$

□

Definition 9. A PAN is in *parallel normal form* if it is in normal form and the set of rules Δ satisfies

$$(X_1\|\ldots\|X_k \to t_1.t_2) \in \Delta \ \wedge\ t_1 \succ^{\Delta} \epsilon \implies (X_1\|\ldots\|X_k \to t_2) \in \Delta$$

Lemma 10. *Let Δ be a PAN that is in normal form, but not in parallel normal form. Then there is a rule $(t \to t_1.t_2) \in \Delta$ s.t. $(t \to t_2) \notin \Delta$ and $t_1 \succ^{\Delta}_{par} \epsilon$.*

Proof By Definition 9 there is a rule $(t' \to t'_1.t'_2) \in \Delta$ s.t. $(t' \to t'_2) \notin \Delta$ and $t'_1 \succ^{\Delta} \epsilon$. Let σ be a sequence of rule applications s.t. $t'_1 \xrightarrow{\sigma} \epsilon$ and σ contains a minimal number of seq-rules. There are two cases:

1. If there are no seq-rules in σ then all rules in σ are par-rules. Thus $t'_1 \succ^{\Delta}_{par} \epsilon$ and the proof is complete.
2. If there are seq-rules in σ then let $(t \to t_1.t_2)$ be the last seq-rule that is applied in σ. It follows that t_1 can be reduced to ϵ by using par-rules only. Thus $t_1 \succ^{\Delta}_{par} \epsilon$. Also $(t \to t_2) \notin \Delta$, because the number of seq-rules in σ is minimal.

□

Lemma 11. *For every PAN Δ in normal form an equivalent (w.r.t. reachability) PAN Δ' in parallel normal form can be effectively constructed.*

Proof We start with $\Delta' := \Delta$ and add new rules $(X_1\|\ldots\|X_k \to t_2)$ to Δ' if $(X_1\|\ldots\|X_k \to t_1.t_2) \in \Delta'$ and $t_1 \succ^{\Delta'}_{par} \epsilon$.

It will turn out that it suffices to check $t_1 \succ^{\Delta'}_{par} \epsilon$ instead of $t_1 \succ^{\Delta'} \epsilon$, because of Lemma 10. Let

$$NF(t \to t') := \begin{cases} \{t \to t'\}, & \text{if } t' = t'_1.t'_2 \\ \{t \to t'\}, & \text{if } t' \in \mathcal{T}_{par} \\ \{t \to Y_1\|\ldots\|Y_k,\, Y_1 \to t'_1, \ldots, Y_k \to t'_k\}, & \text{if } t' = t'_1\|\ldots\|t'_k \end{cases}$$

where no t'_i is a parallel composition and all Y_i are new variables. This function transforms a rule into normal form (see Def. 5). Now we can describe the algorithm that constructs Δ'.

$\Delta' := \Delta; \mathit{flag} := \mathit{true};$
While flag do
  flag := false;
  For every $(t \to t_1.t_2) \in \Delta'$ with $(t \to t_2) \notin \Delta'$ do
    If $\mathcal{E}_{\Delta'}(t_1)$ then $\Delta' := \Delta' \cup NF(t \to t_2)$, $\mathit{flag} := \mathit{true}$ fi;
  od;
od;

Where $\mathcal{E}_{\Delta'}$ is the function from Lemma 8.

1. The algorithm terminates. Let

$$m_i := |\{(t \to t_1.t_2) \in \Delta' \mid size(t_1.t_2) = i \ \wedge \ t_1 \succ^{\Delta} \epsilon \ \wedge \ (t \to t_2) \notin \Delta'\}|$$

   Let k be the largest i s.t. $m_i \neq 0$. In every loop $(m_k, \ldots, m_1)$ decreases w.r.t. lexicographic order.
2. The result Δ' is in parallel normal form. Firstly Δ' is in normal form, because Δ is in normal form and the function NF is applied to every rule that is added to Δ'. Secondly at the end Δ' does not contain any rule $t \to t_1.t_2$ with $t_1 \succ^{\Delta'}_{par} \epsilon$ and $(t \to t_2) \notin \Delta'$. By Lemma 10 it doesn't contain a rule $t' \to t'_1.t'_2$ with $t'_1 \succ^{\Delta'} \epsilon$ and $(t' \to t'_2) \notin \Delta'$ either.

□

Proposition 12. *Let Δ be a PAN in normal form and Δ' the PAN that was constructed by the algorithm in Lemma 11. A state consisting only of variables in Var (no Y_i) is reachable in Δ if and only if it is reachable in Δ'.*

Now we can finally prove the decidability of the reachability problem. It is crucial that the PAN is in parallel normal form.

Lemma 13. *Let Δ be a PAN in parallel normal form, t_0 the initial state and t a state. It is decidable if $t_0 \succ^{\Delta} t$.*

Proof by induction on lexicographically ordered pairs $(size(t), size(t_0))$.
The base case $(0, 0)$ is trivial. Otherwise there are four cases:

1. $t_0 = t_1.t_2$ and $t = t_3 \| t_4$. $t_1.t_2 \succ t_3 \| t_4$ iff $t_1 \succ \epsilon$ and $t_2 \succ t_3 \| t_4$. Both are decidable by induction hypothesis.
2. $t_0 = t_1.t_2$ and $t = t'_1.t'_2.\ldots.t'_k$ s.t. no t'_i is a sequential composition. Then $t_1.t_2 \succ t'_1.t'_2.\ldots.t'_k$ iff $\exists 0 \leq j < k.\ t_1 \succ t'_1.\ldots.t'_j \ \wedge \ t_2 = t'_{j+1} \ldots t'_k$. This is decidable by induction hypothesis. Note that j can be 0. In this case $t_1 \succ \epsilon$. j cannot be equal to k, because t_2 cannot change if t_1 is not reduced to ϵ.
3. In this case the outermost operator in t_0 is parallel composition, while the outermost operator in t is sequential composition. There are two cases:
   (a) The operator for sequential composition occurs in t_0. Thus $t_0 = t_1 \| (t_2.t_3)$ and $t = t'_1.t'_2$.

$$\begin{aligned} t_1 \| (t_2.t_3) \succ t'_1.t'_2 \iff & (t_1 \succ t'_1.t'_2 \ \wedge \ t_2.t_3 \succ \epsilon) \ \vee \\ & (t_1 \succ \epsilon \ \wedge \ t_2.t_3 \succ t'_1.t'_2) \ \vee \\ & (t_2 \succ \epsilon \ \wedge \ t_1 \| t_3 \succ t'_1.t'_2) \end{aligned}$$

   This is decidable by induction hypothesis.

(b) The operator for sequential composition does not occur in t_0. Thus $t_0 \in \mathcal{T}_{par}$ and $t = t_1.t_2$. Let the function $Pre : \mathcal{T} \to 2^{\mathcal{T}}$ be defined by

$$\begin{aligned} Pre(X) &:= \{X\} \\ Pre(t_1.t_2) &:= \{l \mid (l \to r_1.r_2) \in \Delta \ \wedge \ r_1 \succ t_1 \ \wedge \ r_2 = t_2\} \\ Pre(t_1 \| t_2) &:= \{t_1' \| t_2' \mid t_1' \in Pre(t_1) \ \wedge \ t_2' \in Pre(t_2)\} \end{aligned}$$

As $size(t_1) < size(t_1.t_2)$ the induction hypothesis is applicable. Thus $Pre(t_1.t_2)$ is finite and computable, because Δ is finite. As Δ is in parallel normal form it follows that

$$t_0 \succ t_1.t_2 \iff \exists t' \in Pre(t_1.t_2).\ t_0 \succ_{par} t'$$

As $t_0 \in \mathcal{T}_{par}$ and $t' \in \mathcal{T}_{par}$ it is decidable if $t_0 \succ_{par} t'$, because of Proposition 7.

4. Now the outermost operator in both t_0 and t is parallel composition.
   (a) If the operator for sequential composition occurs in t_0, then there are t_1, t_2, t_3 s.t. $t_0 = (t_1.t_2) \| t_3$. Let $t_1' \| t_2' \| \ldots \| t_k' := t$ s.t. no t_i' is a parallel composition.

$$\begin{aligned} (t_1.t_2) \| t_3 \succ t_1' \| t_2' \| \ldots \| t_k' \iff & (t_1 \succ \epsilon \ \wedge \ t_2 \| t_3 \succ t_1' \| t_2' \| \ldots \| t_k') \ \vee \\ & \exists i.\ (t_1.t_2 \succ t_i' \ \wedge \\ & \quad t_3 \succ t_1' \| \ldots \| t_{i-1}' \| t_{i+1}' \| \ldots \| t_k') \end{aligned}$$

   This is decidable by induction hypothesis.

   (b) If no sequential composition occurs in t_0, then $t_0 \in \mathcal{T}_{par}$. By induction hypothesis $Pre(t)$ can be computed. As Δ is in parallel normal form it follows that

$$t_0 \succ t \iff \exists t' \in Pre(t).\ t_0 \succ_{par} t'$$

   This is decidable by Proposition 7, because $Pre(t) \subseteq \mathcal{T}_{par}$ and $Pre(t)$ is finite.

□

Theorem 14. *Let Δ be a PAN, t_0 the initial state and t a state. It is decidable if $t_0 \succ t$.*

Proof First transform Δ into a PAN Δ' in parallel normal form (Lemma 11). By Proposition 12 Δ' is equivalent to Δ w.r.t. reachability. Finally by Lemma 13 reachability is decidable for Δ'. □

The algorithms in Lemma 11 and Lemma 13 use only exponentially many instances of the reachability problem for Petri nets.

4 Extensions

In the previous section the problem was if one given state is reachable. Here we deal with the question if there is a reachable state that has certain properties. First we define a simple logic to describe properties of states, and then we show that it is decidable if a PAN-process can reach a state that satisfies a given property.

Definition 15. Let Act be the set of atomic actions. The syntax of the state-formulae is

$$\Phi ::= a \mid \neg\Phi \mid \Phi_1 \vee \Phi_2 \mid \Phi_1 \wedge \Phi_2$$

where $a \in Act$. The denotation of a formula is defined by

$$\begin{aligned} ||a|| &:= \{t \in \mathcal{T} \mid \exists t'.\ t \xrightarrow{a} t'\} \\ ||\neg\Phi|| &:= \mathcal{T} - ||\Phi|| \\ ||\Phi_1 \vee \Phi_2|| &:= ||\Phi_1|| \cup ||\Phi_2|| \\ ||\Phi_1 \wedge \Phi_2|| &:= ||\Phi_1|| \cap ||\Phi_2|| \end{aligned}$$

So $||\Phi||$ denotes a (possibly infinite) set of PAN-terms. To simplify the notation we use sets of actions. Let $A := \{a_1, \ldots, a_k\} \subseteq Act$ and

$$\begin{aligned} ||A|| &:= ||a_1|| \cap \ldots \cap ||a_k|| \\ ||-A|| &:= ||\neg a_1|| \cap \ldots \cap ||\neg a_k|| \end{aligned}$$

By transformation to disjunctive normal form every state-formula Φ can be written as

$$(A_1^+ \wedge -A_1^-) \vee \ldots \vee (A_n^+ \wedge -A_n^-)$$

where $A_i^+, A_i^- \subseteq Act$. The modal operator $\diamond$ is defined as usual.

$$||\diamond\Phi|| := \{t \mid \exists\sigma.\ t \xrightarrow{\sigma} t' \in ||\Phi||\}$$

Let $t \in \mathcal{T}$ be a PAN-term. For $t \in ||\Phi||$ we also write $t \models \Phi$.

Let there be a PAN Δ in parallel normal form (Def. 9) with initial state t_0 and Φ a state-formula. We now describe a tableau system that decides the problem $t_0 \models \diamond\Phi$. As Φ can be transformed into disjunctive normal form and $t \models \diamond(\Phi_1 \vee \Phi_2) \iff t \models \diamond(\Phi_1) \vee t \models \diamond(\Phi_2)$ it suffices to show decidability for formulae of the form $\diamond(A^+ \wedge -A^-)$. The nodes in the tableau will be sets of formulae, which will be interpreted conjunctively. Γ denotes a set of formulae. The branches are interpreted disjunctively. The tableau is successful iff there is a successful leaf.

For technical reasons we introduce a new operator ∇ that is defined by

$$||\nabla\Phi|| := \{t \mid \exists\sigma, t' \neq \epsilon.\ t \xrightarrow{\sigma} t' \in ||\Phi||\}$$

Now we define the tableau rules. Every node in the tableau consists of a set of formulae. We describe the transformations of single elements of these sets. So the formula $\cup\Gamma$ should be appended to every node, where Γ denotes a set of formulae. We leave this out to simplify the notation.

$$SEQ1 \quad \frac{\{t_1.t_2 \vdash \Diamond(A^+ \wedge -A^-)\}}{\{t_1 \vdash \nabla(A^+ \wedge -A^-)\} \quad \{t_1 \succ \epsilon, t_2 \vdash \Diamond(A^+ \wedge -A^-)\}}$$

$$SEQ2 \quad \frac{\{t_1.t_2 \vdash \nabla(A^+ \wedge -A^-)\}}{\{t_1 \vdash \nabla(A^+ \wedge -A^-)\} \quad \{t_1 \succ \epsilon, t_2 \vdash \nabla(A^+ \wedge -A^-)\}}$$

$$SP1 \quad \frac{\{(t_1.t_2)\|t_3 \vdash \Diamond(A^+ \wedge -A^-)\}}{\{t_1 \succ \epsilon, t_2\|t_3 \vdash \Diamond(A^+ \wedge -A^-)\} \quad \ldots\{t_1 \vdash \nabla(A_1^+ \wedge -A^-), t_3 \vdash \Diamond(A_2^+ \wedge -A^-)\}\ldots}$$

where $A^+ = A_1^+ \cup A_2^+$

$$SP2 \quad \frac{\{(t_1.t_2)\|t_3 \vdash \nabla(A^+ \wedge -A^-)\}}{\{t_1 \succ \epsilon, t_2\|t_3 \vdash \nabla(A^+ \wedge -A^-)\} \quad \ldots\{t_1 \vdash \nabla(A_1^+ \wedge -A^-), t_3 \vdash \Diamond(A_2^+ \wedge -A^-)\}\ldots}$$

where $A^+ = A_1^+ \cup A_2^+$

$$PAR1 \quad \frac{\{t \vdash \Diamond(A^+ \wedge -A^-)\}}{\ldots\{r_1^1 \vdash \nabla(A_1^+ \wedge -A^-), \ldots, r_k^1 \vdash \nabla(A_k^+ \wedge -A^-)\}\ldots}$$

where $t \in \mathcal{T}_{par}$ and $(l_i \rightarrow r_i^1.r_i^2) \in \Delta$, for $i = 1, \ldots, k$

$\exists t' \in \mathcal{T}_{par}.\ t \succ_{par} (t'\|l_1\|\ldots\|l_k)$ with $t' \models (A_0^+ \wedge -A^-)$

and $A^+ = A_0^+ \cup A_1^+ \cup \ldots \cup A_k^+$

$$PAR2 \quad \frac{\{t \vdash \nabla(A^+ \wedge -A^-)\}}{\ldots\{r_1^1 \vdash \nabla(A_1^+ \wedge -A^-), \ldots, r_k^1 \vdash \nabla(A_k^+ \wedge -A^-)\}\ldots}$$

where $t \in \mathcal{T}_{par}$ and $(l_i \rightarrow r_i^1.r_i^2) \in \Delta$, for $i = 1, \ldots, k$

$\exists t' \in \mathcal{T}_{par}.\ t \succ_{par} (t'\|l_1\|\ldots\|l_k)$ with $t' \models (A_0^+ \wedge -A^-)$

and $A^+ = A_0^+ \cup A_1^+ \cup \ldots \cup A_k^+$, and $(k \neq 0 \vee t' \neq \epsilon)$

$$E \quad \frac{\{t \succ \epsilon\} \cup \Gamma}{\Gamma}$$

if $t \succ \epsilon$

The dots in the rules SP1 and SP2 symbolize all different (but only finitely many) ways of partitioning A^+ into A_1^+ and A_2^+. In PAR1 and PAR2 the dots symbolize all different ways of choosing k, the rules $(l_i \rightarrow r_i^1.r_i^2)$ and the partitioning of A^+ into $A_0^+, \ldots, A_k^+$. Again there are only finitely many.

Lemma 16. *If the side conditions of a rule are satisfied, then the antecedent of a rule is true iff one of its consequents is true.*

Proof For the rules $SEQ1, SEQ2, SP1, SP2$ and E this follows directly from the definitions.

PAR1 and PAR2 are true, because Δ is in parallel normal form. The only difference between PAR1 and PAR2 is the condition $k \neq 0 \vee t' \neq \epsilon$. This ensures that the reachable state that satisfies $(A^+ \wedge -A^-)$ is not ϵ. □

Definition 17. Termination conditions: A node marked with a set of formulae Γ is a terminal node iff one of the following conditions is satisfied.

1. Γ is empty.
2. $\Gamma = \Gamma' \cup \{t \succ \epsilon\}$ for some $t \in \mathcal{T}$ and **not** $t \succ \epsilon$.
3. The same node Γ occurred earlier on the path from the root to the current node.

Terminals of type 1 are successful, while terminals of types 2 and 3 are unsuccessful.

Lemma 18. *The tableau for a given root can be effectively constructed.*

Proof

1. The side conditions of the rules are decidable: For rule E this follows from Theorem 14. For $PAR1$ and $PAR2$ this follows from the decidability of the submarking reachability problem for Petri nets, because all l_i are in $\mathcal{T}_{par}$ and Δ is finite.
2. The tableau is finitely branching: For SEQ1 and SEQ2 this is trivial. For SP1 and SP2 this is because there are only finitely many different ways to partition A^+ into A_1^+ and A_2^+. For PAR1 and PAR2 this is because Δ is finite.
3. The tableau is finite: Let t_0 be the state in the root-node. There are only finitely many different subterms of t_0. As Δ is finite there are only finitely many different subterms r_i^1 of right-hand sides of rules in Δ. Only finitely many different formulae of the form $\Diamond(A^+ \wedge -A^-)$ or $\nabla(A^+ \wedge -A^-)$ can occur in the tableau. Therefore there are only finitely many different nodes in the tableau. Thus the construction of the tableau must terminate, because of termination condition 3.

□

Now we prove the soundness and completeness of the tableau system.

Lemma 19. *If there is a successful tableau with root $t \vdash \Diamond(A^+ \wedge -A^-)$, then $t \models \Diamond(A^+ \wedge -A^-)$.*

Proof If the tableau is successful, then it has a branch that ends with a successful (empty) node. This node is certainly true. By Lemma 16 the root-node must be true as well. □

Lemma 20. *Let $Op \in \{\Diamond, \nabla\}$. Let there be a node of the form $\{t \vdash Op(A^+ \wedge -A^-)\} \cup \Gamma$, s.t. $t \models Op(A^+ \wedge -A^-)$ and Γ is true.*

Then the tableau with this root has a branch leading to a node Γ.

Proof by complete induction on lexicographically ordered pairs (x, y) s.t. $(x, y) := (length(\sigma), size(t))$, where σ is a sequence of minimal length s.t. $t \xrightarrow{\sigma} t'$ and $t' \models (A^+ \wedge -A^-)$ (and $t' \neq \epsilon$ if $Op = \nabla$). Such a sequence must exist, because $t \models Op(A^+ \wedge -A^-)$.

If $(x, y) = (0, 0)$ then $t = \epsilon$ and $A^+ = \{\}$. The rule PAR1 is applicable and the one child-node is Γ.

Apply the rules for the construction of the tableau. By Lemma 16 at least one child-node must be true. Choose the true child-node that corresponds to σ.

For the rules SEQ1, SEQ2, SP1 and SP2 in the child node x is lower or equal and y is smaller. With the induction hypothesis and rule E the result follows.

For the rules PAR1 and PAR2 the second component y may have increased in the child-node, but the first component x is always smaller. By multiple application of the induction hypothesis the result follows.

This construction cannot be interrupted by termination condition 3, because this would contradict the minimality of the length of σ. $\square$

Corollary 21. *If $t \models \Diamond(A^+ \wedge -A^-)$, then there is a successful tableau with root $t \vdash \Diamond(A^+ \wedge -A^-)$.*

Proof By Lemma 20 there is a branch in the tableau with root $t \vdash \Diamond(A^+ \wedge -A^-)$ that ends with the empty node. By termination condition 1 this node is a successful terminal. Therefore the tableau is successful. $\square$

Theorem 22. *Let there be a PAN Δ with initial state t_0 and Φ a state-formula.*

It is decidable if $t_0 \models \Diamond\Phi$.

Proof By the algorithm in Lemma 11 the PAN Δ can be transformed into an equivalent PAN Δ' in parallel normal form. It is clear that $\Delta \subseteq \Delta'$. The new rules in $\Delta' \backslash \Delta$ will all be labelled with the new atomic action τ, where $\tau \notin Act$. (So all rules in $\Delta' \backslash \Delta$ are of the form $t \xrightarrow{\tau} t'$). It follows that $t_0 \models \Diamond\Phi$ w.r.t. Δ' iff $t_0 \models \Diamond\Phi$ w.r.t. Δ.

It suffices to show decidability for formulae of the form $\Diamond(A^+ \wedge -A^-)$. As Δ' is in parallel normal form we can apply the tableau system. By Lemma 18 the tableau with root $t_0 \vdash \Diamond(A^+ \wedge -A^-)$ can be effectively constructed. It follows from Lemma 19 and Corollary 21 that $t_0 \models \Diamond(A^+ \wedge -A^-)$ if and only if the tableau is successful. $\square$

This result can also be used to decide the reachability of a state of deadlock. Let Δ be a PAN with initial state t_0 and A the set of variables used in Δ. A state of deadlock is reachable iff $t_0 \models \Diamond(-A)$.

5 Example

In this section we show how PAN-processes can be used to model a simple process management system. It consists of a master manager X and possibly several local submanagers Y. There are also two terminal processes P_1 and P_2 and a counter C. Let

$$Var := \{X, Y, Z, C, P_1, P_2, T\}$$

and

$$Act := \{start, startjob, newP_1, newP_2, a, b, shutdown, term, abort, z, c\}$$

and the following set of rules Δ.

$$X \xrightarrow{start} (Y.Z)\|X\|C \tag{1}$$

$$Y \xrightarrow{startjob} \epsilon \tag{2}$$

$$Y \xrightarrow{newP_1} Y.P_1.Y \tag{3}$$

$$Y \xrightarrow{newP_2} Y.P_2.Y \tag{4}$$

$$P_1 \xrightarrow{a} \epsilon \tag{5}$$

$$P_2 \xrightarrow{b} \epsilon \tag{6}$$

$$Z\|X\|C \xrightarrow{term} X \tag{7}$$

$$X \xrightarrow{shutdown} T \tag{8}$$

$$Z\|T \xrightarrow{abort} T \tag{9}$$

$$Z \xrightarrow{z} Z \tag{10}$$

$$C \xrightarrow{c} C \tag{11}$$

1. In this rule the master manager starts a new submanager. All submanagers run in parallel. Variable C is used to count the number of running submanagers.
2. In this rule the submanager Y terminates and thus enables the execution of a terminal process (P_1 or P_2). After this process terminates the submanager becomes active again.
3. Every submanager stores its jobs in a stack (last-in first-out). Here a process P_1 is added to the stack.
4. Here a process P_2 is added to the stack.
5. This rule indicates the run of process P_1.
6. This rule indicates the run of process P_2.
7. A regular termination of a submanager. This reduces the counter C by 1.

8. This rule indicates the shutdown of the master manager. After the shutdown no new submanagers can be created. All submanagers that are still running now terminate irregularly. Variable C is now used to count the number of these irregular terminations.
9. An irregular termination of a submanager (after the shutdown of the master manager). Note that unlike in the regular termination the number of variables C is not reduced.
10. This rule changes nothing. It only indicates that variable Z is present and active.
11. This rule changes nothing. It only indicates that variable C is present and active.

Some properties of this system can be verified with the algorithms of the two previous sections:

$$X \succ T||C^5$$

This means that it is possible to reach a final state where exactly 5 submanagers have terminated irregularly.

$$X \models \Diamond(c \wedge -(Act\backslash\{c\}))$$

It is possible to reach a state where no action but c is possible. So in this state the master manager has been shut down and all submanagers have terminated, at least one of them irregularly.

$$X \not\models \Diamond(abort \wedge \neg c)$$

An irregular termination is not possible if no variable C is present. (C is used to count submanagers).

6 Conclusion

We have defined PAN-processes, a very expressive model of infinite state concurrent systems that subsumes Petri nets and PA-processes.

It extends Petri nets by introducing an operator for sequential composition. This can be seen as the possibility to invoke subroutines. PAN-processes also extends PA-processes by allowing for synchronization. PAN-processes are a very expressive model for concurrent systems, as even slight generalizations would make it Turing-powerful. Even so, many temporal logics (EF, CTL, LTL, linear time μ-calculus, modal μ-calculus) are undecidable for it [Esp97, Esp96, BH96, EK95].

We have shown that the reachability problem is still decidable for PAN-processes. Moreover, it is even decidable if there is a reachable state that satisfies some properties that can be described in a certain simple logic. This is a generalization of the decidability of the submarking reachability problem for Petri nets. PAN-processes can be used to model systems that exceed the bounds of the expressiveness of Petri nets.

Acknowledgements I would like to thank Javier Esparza and Astrid Kiehn for helpful discussions.

References

[BEH95] A. Bouajjani, R. Echahed, and P. Habermehl. Verifying infinite state processes with sequential and parallel composition. In *Proceedings of POPL'95*, pages 95–106. ACM Press, 1995.

[BH96] A. Bouajjani and P. Habermehl. Constrained properties, semilinear systems, and petri nets. In Ugo Montanari and Vladimiro Sassone, editors, *Proceedings of CONCUR'96*, number 1119 in LNCS. Springer Verlag, 1996.

[BS94] O. Burkart and B. Steffen. Pushdown processes: Parallel composition and model checking. In *CONCUR'94*, number 836 in LNCS, pages 98–113. Springer Verlag, 1994.

[Chr93] S. Christensen. *Decidability and Decomposition in Process Algebras*. PhD thesis, Edinburgh University, 1993.

[EK95] J. Esparza and A. Kiehn. On the model checking problem for branching time logics and basic parallel processes. In *CAV'95*, number 939 in LNCS, pages 353–366. Springer Verlag, 1995.

[Esp96] J. Esparza. More infinite results. In B. Steffen and T. Margaria, editors, *Proceedings of INFINITY'96*, number MIP-9614 in Technical report series of the University of Passau. University of Passau, 1996.

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

[Kuc96] A. Kucera. Regularity is decidable for normed pa processes in polynomial time. In *Foundations of Software Technology and Theoretical Computer Science (FSTTCS'96)*, number 1180 in LNCS. Springer Verlag, 1996.

[May84] E. Mayr. An algorithm for the general petri net reachability problem. *SIAM Journal of Computing*, 13:441–460, 1984.

[May96a] Richard Mayr. Model checking pa-processes. Technical Report TUM-I9640, TU-München, December 1996. To appear in CONCUR'97.

[May96b] Richard Mayr. Weak bisimulation and model checking for basic parallel processes. In *Foundations of Software Technology and Theoretical Computer Science (FSTTCS'96)*, number 1180 in LNCS. Springer Verlag, 1996.

[May97] Richard Mayr. Tableau methods for pa-processes. In *Analytic Tableaux and Related Methods (TABLEAUX'97)*, LNAI. Springer Verlag, 1997. to appear.