Model Checking
PA-Processes

Richard Mayr

TUM-19640
SFB-Bericht Nr. 342/17/96 A
November 1996
Alle Rechte vorbehalten
Nachdruck auch auszugsweise verboten

©1996 SFB 342 Methoden und Werkzeuge für
die Nutzung paralleler Architekturen

Anforderungen an: Prof. Dr. A. Bode
Sprecher SFB 342
Institut für Informatik
Technische Universität München
D-80290 München, Germany

Druck: Fakultät für Informatik der
Technischen Universität München
Model Checking 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

PA (Process algebra) 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). They are a simple model
for infinite state concurrent systems.

We show that the model checking problem for the branching time temporal
logic $EF$ is decidable for PA-processes.

Keywords: PA-processes, model checking, process algebras, tableau systems

1 Introduction

The Process Algebra PA is a simple model of infinite state concurrent systems. It has
operators for nondeterministic choice, parallel composition, sequential composition
and recursion. PA-processes and Petri nets are incomparable, meaning that neither
model is more expressive than the other one. Unlike BPPs, PA is not a syntactical
subset of CCS [Mil89], because CCS does not have an explicit operator for sequential
composition. However, as CCS can simulate sequential composition by parallel
composition and synchronization, PA is still a weaker model than CCS. PA-processes
are a superset of both Basic Parallel Processes (BPP) [Chr93] and context-free
processes (BPA).

Here we study the model checking problem for PA-processes. This is the problem
of deciding if a given PA-process satisfies a property coded as a formula in a certain
temporal logic.

For BPPs the situation is already fairly clear. It has been shown in [EK95]
that the model checking problem for BPPs is undecidable for the branching time
temporal logic $EG$, whose formulae are built out of the boolean operators, $EX$ (for
some successor) and $EG$ (for some path always in the future). On the other hand
the model checking problem is decidable for the logic $EF$, that uses the boolean
operators, and the temporal operators $EX$ and $EF$ (for some path eventually in the
Therefore, the logic EF (also called UB$^{-}$ in [Esp]), seems to be the largest branching time logic with a decidable model checking problem. The model checking problem for BPPs and EF (UB$^{-}$) is PSPACE-complete [May96a, May96b].

Here we show that the model checking problem with the logic EF is decidable even for PA-processes. In section 2 we define PA-processes. In section 3 we describe the tableau system the solves the model checking problem, while in section 4 we prove its soundness and completeness. Section 5 describes a possible extension of the logic by adding constraints on sequences of actions. The paper closes with a section on open problems and related work.

2 PA-Processes

The definition of PA is as follows: Assume a countably infinite set of atomic actions $Act = \{a, b, c, \ldots\}$ and a countably infinite set of process variables $Var = \{X, Y, Z, \ldots\}$. The class of PA expressions is defined by the following abstract syntax

$$E ::= \epsilon | X | aE | E + E | E \parallel E | E.E$$

A PA is defined by a family of recursive equations $\{X_i := E_i | 1 \leq i \leq n\}$, where the $X_i$ are distinct and the $E_i$ are PA expressions at most containing the variables $\{X_1, \ldots, X_n\}$. We assume that every variable occurrence in the $E_i$ is guarded, i.e. appears within the scope of an action prefix, which ensures that PA-processes generate finitely branching transition graphs. This would not be true if unguarded expressions were allowed. For example, the process $X := a + a \parallel X$ generates an infinitely branching transition graph. For every $a \in Act$ the transition relation $\rightarrow_a$ is the least relation satisfying the following inference rules:

$$aE \rightarrow_a E \quad \frac{E \rightarrow E'}{E + F \rightarrow F'} \quad \frac{F \rightarrow F'}{E + F \rightarrow F'} \quad \frac{E \rightarrow E'}{X \rightarrow X'}(X := E)$$

Alternatively, PA-processes can be represented by a state described by a term of the form

$$G ::= \epsilon | X | G_1.G_2 | G_1 \parallel G_2$$

and set of rules $\Delta$ of the form $X \rightarrow G$ whose application to states must respect sequential composition. This is described by the following inference rules:

$$X \rightarrow G \quad \text{if} \ (X \rightarrow G) \in \Delta$$

We assume w.r.t. that for every variable $X$ there is at least one rule $X \rightarrow t$. The transition relation $\rightarrow$ is extended to sequences of actions $\rightarrow$ in the standard way. If the sequence $\sigma$ is of no account, then we just write $\rightarrow$. 

2
BPPs are the subset of PA-processes without sequential composition, while context-free processes are the subset of PA-processes without parallel composition. Unlike for PA-processes there is a one-to-one correspondence between BPPs and a class of labelled Petri nets, the communication-free nets [Esp]. In these nets every transition has exactly one input place with an arc labelled by 1.

3 The Tableau System

Model checking algorithms can be divided into two classes: iterative algorithms and tableau-based algorithms. The iterative algorithms compute all the states of the system which have the desired property, and usually yield higher efficiency in the worst case. The tableau-based algorithms are designed to check whether a particular expression has a temporal property. This is called local model checking which avoids the investigation of irrelevant parts of the process being verified. Therefore this method is applicable for the verification of systems with infinite state spaces. In local model checking the proof system is developed in a goal directed fashion (top down). A property holds iff there is a proof tree with a successful leaf which witnesses this truth. The algorithm for the following problem is tableau-based and decides the truth of an EF-formula for a PA-process by examining only finitely many states.

3.1 The Temporal Logic EF

The branching time temporal logic EF of [Esp, Esp96] is used to describe properties of PA-processes. We fix a countably infinite set of atomic actions Act. The syntax of the calculus is as follows:

$$\Phi \overset{\text{def}}{=} a \mid \neg \Phi \mid \Phi_1 \land \Phi_2 \mid \Diamond \Phi$$

where $a \in Act$ ranges over atomic actions. For convenience disjunction and another modal operator $\square$ can be added by defining $\square := \neg \Diamond \neg$.

Let $\mathcal{F}$ be the set of all EF-formulae. Let $\Omega$ be the set of all processes in the process algebra. The denotation $\|\Phi\|$ of a formula $\Phi$ is the set of processes inductively defined by the following rules:

$$\begin{align*}
\|a\| &= \{ t \mid \exists t' \xrightarrow{a} t' \} \\
\|\neg \Phi\| &= \Omega - \|\Phi\| \\
\|\Phi_1 \land \Phi_2\| &= \|\Phi_1\| \cap \|\Phi_2\| \\
\|\Diamond \Phi\| &= \{ t \mid \exists t', t' \in \|\Phi\| \}
\end{align*}$$

The property $t \in \|\Phi\|$ is also denoted by $t \models \Phi$. An instance of the model checking problem is a PA process algebra, a term $t$ in the algebra and an EF-formula $\Phi$. The question is if $t \models \Phi$.

In order to simplify the presentation we have left out the one-step nexttime-operator $EX$ for now. In Section 5 we’ll show that it can be added to the logic.
without causing any problems. In this framework this operator is often denoted by \([a]\), with \(a \in \text{Act}\) and defined by

\[
\| [a] \Phi \| = \{ t \mid \exists t \xrightarrow{a} t' \in \| \Phi \| \}
\]

The decidability results carry over to the logic that includes the nexttime-operator (see Section 5).

While the model checking problem with EF is undecidable for general Petri nets [Esp], it is decidable and PSPACE-complete for BPPs [May96a, May96b]. Here we show that model checking with the logic EF is decidable for PA-processes.

**Definition 3.1** \(\mathcal{F}_d \subset \mathcal{F}\) is defined as the set of all EF-formulae with a nesting-depth of modal operators \(\Diamond\) of at most \(d\). (It follows that formulae in \(\mathcal{F}_0\) contain no modal operators.)

In order to simplify the notation we use some abbreviations:

Let \(A = \{ a_1, \ldots, a_n \} \subset \text{Act}\) be a set of atomic actions, then

\[ t \models A \iff t \models a_1 \land \ldots \land a_n \]

and

\[ t \models \neg A \iff t \models \neg a_1 \land \ldots \land \neg a_n \]

The decidability proof of the model checking problem is done by induction on the nesting depth \(d\) of modal operators in the formula. For a term \(t\) and a formula \(\Phi \in \mathcal{F}_d\) the algorithm builds a finite tableau for \(t \models \Phi\) by using properties of the form \(t' \models F'\) with \(F' \in \mathcal{F}_{d-1}\) as side conditions.

First we reduce the problem to a simpler form.

**Definition 3.2** The set of conjunctive formulae \(\mathcal{F}^c \subset \mathcal{F}\) is the smallest set of formulae satisfying the following conditions:

1. \(A^+ \land \neg A^-\) is a conjunctive formula for \(A^+, A^- \subset \text{Act}\)

2. \(A^+ \land \neg A^- \land \bigwedge_{i \in I} \Psi_i \land \bigwedge_{i \in J} \neg \Diamond \gamma_j\) is a conjunctive formula if \(A^+, A^- \subset \text{Act}\) and \(\Psi_i, \gamma_j \in \mathcal{F}^c\).

Let \(\mathcal{F}_d^c := \mathcal{F}_d \cap \mathcal{F}^c\).

A formula \(\Phi\) is in normal form if \(\Phi = \bigvee_{i \in I} \Diamond \Psi_i\) s.t. the \(\Psi_i\) are conjunctive formulae. \(\mathcal{F}_d^c \subset \mathcal{F}_d\) are the formulae in normal form in \(\mathcal{F}_d\).

**Lemma 3.3** Any EF-formula \(\Phi = \Diamond \Psi\) is equivalent to a formula in normal form.

**Proof** By induction on the nesting-depth \(d\) of modal operators in \(\Psi\).

1. If \(d = 0\) then \(\Psi\) doesn’t contain any modal operators, so it can be transformed into disjunctive normal form \(\bigvee_{i \in I} A_i^+ \land \neg A_i^-\). Therefore \(\Phi\) is equivalent to \(\bigvee_{i \in I} \Diamond (A_i^+ \land \neg A_i^-)\). This is a formula in normal form.
2. Now \( d > 0 \). By induction hypothesis we can transform all subformulae \( \Diamond \varphi \) of \( \Psi \) into normal form, obtaining a formula \( \Psi' \). Then transform \( \Psi' \) into disjunctive normal form \( \Psi'' = \bigvee_{i \in I} \gamma_i \). Thus \( \Phi \) is equivalent to \( \Phi' = \Diamond \left( \bigvee_{i \in I} \gamma_i \right) = \bigvee_{i \in I} \Diamond \gamma_i \). This is in normal form, because all \( \gamma_i \) are conjunctive formulae.

\[ \square \]

**Lemma 3.4** Every model checking problem for EF is decidable iff it is decidable for all formulae \( \Diamond \Phi \) with \( \Phi \in \mathcal{F}^c \).

**Proof** If it is decidable for formulae of the form \( \Diamond \Psi \) with \( \Psi \in \mathcal{F}^c \), then it is decidable for formulae in normal form and thus by Lemma 3.3 for all formulae of the form \( \Diamond \Phi \). Simple boolean operations yield the decidability of the whole model checking problem. The other direction is trivial. \[ \square \]

In the sequel all EF-formulae will be conjunctive formulae. Let \( \Phi \in \mathcal{F}^c_d \). Then \( \Diamond \Phi \) has the form \( \Diamond (A^+ \land A^- \land \bigwedge_{i \in I} \Diamond \Psi_i \land \bigwedge_{j \in J} \neg \Diamond \gamma_j) \) where \( A^+ \subseteq \text{Act}, A^- \subseteq \text{Act} \) and \( \Psi_i \in \mathcal{F}^c_{d-1} \) and \( \gamma_j \in \mathcal{F}^c_{d-1} \).

**Remark 3.5** In the definition of PA algebras we assumed that every occurring variable is defined. It follows that in the other representation there is at least one rule \( X \xrightarrow{e} G \) in \( \Delta \) for every \( X \). Therefore a PA-process cannot perform any action if and only if it is empty. This means that \( t \models \Diamond (\neg \text{Act}) \iff \exists e \in e \).

### 3.2 Decomposition

For the construction of a finite tableau that solves the model checking problem it is necessary to split the problem into several smaller subproblems. We do this by showing that properties of a PA-process can be expressed by properties of its sub-processes.

**Lemma 3.6** Let \( t_1, t_2 \) be PA-terms and \( \Phi \) in \( \mathcal{F}^c_d \). There is a set \( I \) and terms \( \Phi_i^1, \Phi_i^2 \in \mathcal{F}^c_{d-1} \) s.t.

\( t_1 \parallel t_2 \models \Diamond \Phi \iff \bigvee_{i \in I} t_1 \models \Diamond \Phi_i^1 \land t_2 \models \Diamond \Phi_i^2 \)

**Proof** \( \Diamond \Phi = \Diamond (A^+ \land A^- \land \bigwedge_{i \in I} \Diamond \Psi_i \land \bigwedge_{j \in J} \neg \Diamond \gamma_j) \) with \( A^+, A^- \subseteq \text{Act}, \Psi_i, \gamma_j \in \mathcal{F}^c_{d-1} \). The proof is done by induction on \( d \).

\( t_1 \parallel t_2 \models \Diamond (A^+ \land A^- \land \bigwedge_{i \in I} \Diamond \Psi_i \land \bigwedge_{j \in J} \neg \Diamond \gamma_j) \)

By definition of EF this is equivalent to

\[ \exists A_1^+ \cup A_2^+ = A^+. \exists t_1 \xrightarrow{e_1} t_1', t_2 \xrightarrow{e_2} t_2'. \ t_1' \models (A_1^+ \land A^-) \land t_2' \models (A_2^+ \land A^-) \land \bigwedge_{i \in I} t_1' \parallel t_2' \models \Diamond \Psi_i \land \bigwedge_{j \in J} t_1' \parallel t_2' \models \neg \Diamond \gamma_j. \]
By induction hypothesis there are $K_i, L_j$ and $\phi_{i,k}^1, \phi_{i,k}^2, \delta_{i,j}, \delta_{i,j}^2 \in \mathcal{F}_{d-1}$ s.t. the expression is equivalent to

$$\exists A_1^+ \cup A_2^+ = A^+. \exists t_1 \rightarrow t_1', t_2 \rightarrow t_2', t_1' \models (A_1^+ \land -A^-) \land t_2' \models (A_2^+ \land -A^-) \land$$

$$\bigwedge_{i \in I, k \in K_i} \exists t_1 \rightarrow t_1', t_2 \rightarrow t_2', t_1' \models \diamond \phi_{i,k}^1 \land t_2' \models \diamond \phi_{i,k}^2 \land \bigwedge_{j \in J, i \in I} \neg (\bigvee_{t_1' \models \diamond \delta_{i,j}^1} \land t_2' \models \neg \diamond \delta_{i,j}^2)$$

By De Morgan this is equivalent to

$$\exists A_1^+ \cup A_2^+ = A^+. \exists t_1 \rightarrow t_1', t_2 \rightarrow t_2', t_1' \models (A_1^+ \land -A^-) \land t_2' \models (A_2^+ \land -A^-) \land$$

$$\bigwedge_{i \in I, k \in K_i} \exists t_1 \rightarrow t_1', t_2 \rightarrow t_2', t_1' \models \diamond \phi_{i,k}^1 \land t_2' \models \diamond \phi_{i,k}^2 \land \bigwedge_{j \in J, i \in I} \neg (\bigvee_{t_1' \models \neg \diamond \delta_{i,j}^1} \land t_2' \models \neg \diamond \delta_{i,j}^2)$$

By transformation to disjunctive normal form we get

$$\exists A_1^+ \cup A_2^+ = A^+. \exists t_1 \rightarrow t_1', t_2 \rightarrow t_2', t_1' \models (A_1^+ \land -A^-) \land t_2' \models (A_2^+ \land -A^-) \land$$

$$\bigwedge_{F:i \rightarrow K_i, G \times H : j \times L_j} \left[ \bigwedge_{i \in I} t_1' \models \diamond \phi_{i,F(i)}^1 \land t_2' \models \diamond \phi_{i,F(i)}^2 \land \bigwedge_{(j,i) \in G \times H} t_1' \models \neg \diamond \delta_{j,i}^1 \land \bigwedge_{(j,i) \in J \times L_j - G \times H} t_2' \models \neg \diamond \delta_{j,i}^2 \right]$$

Here $F$ is a total function $F : I \mapsto \bigcup_{i \in I} K_i$, s.t. $\forall i \in I, F(i) \in K_i$. $G$ and $H$ must satisfy the restriction that if $(j, l) \in G \times H$, then $l \in L_j$. Putting it together again yields

$$\bigvee_{A_1^+ \cup A_2^+ = A^+ \cup F : i \rightarrow K_i, G \times H : j \times L_j} \left[ t_1' \models \diamond (A_1^+ \land -A^-) \land \bigwedge_{i \in I} \diamond \phi_{i,F(i)}^1 \land \bigwedge_{(j,i) \in G \times H} \neg \diamond \delta_{j,i}^1 \land \bigwedge_{(j,i) \in J \times L_j - G \times H} \neg \diamond \delta_{j,i}^2 \right]$$

This is in normal form. ❑

**Lemma 3.7** Let $t_1, t_2$ be PA-terms and $\Phi$ in $\mathcal{F}_d^\omega$. There are sets $N, P, Q$ and terms $\alpha, \beta \in \mathcal{F}_d^\omega$ and $\gamma_p, \delta_q \in \mathcal{F}_{d-1}^\omega$ s.t. $t_1, t_2 \models \diamond \Phi$ iff

$$t_1 \models \diamond (\neg \alpha \lor \beta) \land t_2 \models \diamond \Phi \lor$$

$$t_1 \models \neg \diamond (\neg \alpha \lor \beta) \land t_1 \models \diamond \Phi \lor$$

$$t_1 \models \diamond \alpha \lor \bigvee_{n \in N} \left[ t_1 \models \diamond \beta_n \land \bigwedge_{p \in P(n)} t_2 \models \diamond \gamma_p \land \bigwedge_{q \in Q(n)} (t_2 - \neg \diamond \delta_q) \right]$$

**Proof** by induction on $d$. If $d = 0$ then $\diamond \Phi = \diamond (A^+ \land -A^-)$. The first two cases of the above disjunction are clear. The only remaining case is $t_1 \rightarrow t_1' \neq \epsilon, t_1' \models (A^+ \land -A^-)$. (Here $\exists t_1' \rightarrow \epsilon$.) Choose $\alpha = \text{false}, N = A \land, \beta_a = A^+ \cup \{a\} \land -A^-, P(a) = Q(a) = \emptyset$ for every $a \in A \land$. 

6
Now \( d > 0 \). We can assume that \( \Diamond \Phi \equiv \Diamond (A^+ \land \neg A^- \land \bigwedge_{i \in I} \Diamond \Psi_i \land \bigwedge_{j \in J} \neg \Diamond \gamma_j) \) with \( A^+, A^- \subseteq \text{Act}, \Psi_i, \gamma_j \in \mathcal{F}_{d-1}^\varepsilon \). The first two cases of the above disjunction are obvious. In the third case we have:

\[
\exists t_1 \xrightarrow{*} t'_1 \neq \varepsilon, t'_1 \models A^+ \land \neg A^- \land t'_1 t_2 \models \bigwedge_{i \in I} \Diamond \Psi_i \land \bigwedge_{j \in J} \neg \Diamond \gamma_j
\]

This is equivalent to

\[
t_1 \models \Diamond (A^+ \land \neg A^- \land \neg \Diamond (-A d) \land \bigwedge_{i \in I} \Diamond \Psi_i \land \bigwedge_{j \in J} \neg \Diamond \gamma_j) \lor
\]

\[
t_2 \models \bigwedge_{j \in J} \neg \Diamond \gamma_j \land
\]

\[
\bigvee_{a \in \text{Act}} \exists t_1 \rightarrow t'_1 \left[ t'_1 \models (A^+ \cup \{a\} \land \neg A^-) \land t'_1 \models \bigwedge_{j \in J} \neg \Diamond (\gamma_j \land b) \land t'_1 \models \bigwedge_{i \in I} \Diamond \Psi_i \right]
\]

As \( \Psi_i \in \mathcal{F}_{d-1}^\varepsilon \) there are (by induction hypothesis) \( \alpha_i, \beta_n \in \mathcal{F}_{d-1}^\varepsilon \) and \( \gamma_p, \delta_q \in \mathcal{F}_{d-2}^\varepsilon \) s.t. this is equivalent to

\[
t_1 \models \Diamond (A^+ \land \neg A^- \land \neg \Diamond (-A d) \land \bigwedge_{i \in I} \Diamond \Psi_i \land \bigwedge_{j \in J} \neg \Diamond \gamma_j) \lor
\]

\[
t_2 \models \bigwedge_{j \in J} \neg \Diamond \gamma_j \land
\]

\[
\bigvee_{a \in \text{Act}} \exists t_1 \rightarrow t'_1 \left[ t'_1 \models (A^+ \cup \{a\} \land \neg A^-) \land t'_1 \models \bigwedge_{j \in J} \neg \Diamond (\gamma_j \land b) \land t'_1 \models \bigwedge_{i \in I} \Diamond \Psi_i \right]
\]

\[
\bigwedge_{i \in I} \left[ t'_1 \models \Diamond \beta_n \land \bigwedge_{p \in P(n)} t_2 \models \Diamond \gamma_p \land \bigwedge_{q \in Q(n)} t_2 \models \neg \Diamond \delta_q \right]
\]

This requires some explanation. The case that \( t'_1 \) cannot be reduced to \( \varepsilon \) is already considered in the first line of this formula. So we can assume that \( t'_1 \models \Diamond (-A d) \). Therefore in the application of the induction hypothesis we only need to add the formula \( t_2 \models \Diamond \Psi_i \).

By transformation to disjunctive normal form we get

\[
t_1 \models \Diamond (A^+ \land \neg A^- \land \neg \Diamond (-A d) \land \bigwedge_{i \in I} \Diamond \Psi_i \land \bigwedge_{j \in J} \neg \Diamond \gamma_j) \lor
\]

\[
t_2 \models \bigwedge_{j \in J} \neg \Diamond \gamma_j \land
\]

\[
\bigvee_{a \in \text{Act}} \exists t_1 \rightarrow t'_1 \left[ t'_1 \models (A^+ \cup \{a\} \land \neg A^-) \land t'_1 \models \bigwedge_{j \in J} \neg \Diamond (\gamma_j \land b) \land t'_1 \models \bigwedge_{i \in I} \Diamond \Psi_i \right]
\]

\[
\bigwedge_{i \in I} \left[ t'_1 \models \Diamond \beta_n \land \bigwedge_{p \in P(n)} t_2 \models \Diamond \gamma_p \land \bigwedge_{q \in Q(n)} t_2 \models \neg \Diamond \delta_q \right]
\]

\[
\bigvee_{i \in I, L, F: (i \neq (L \cup U') \rightarrow N_i \text{ } i \in E_L', \bigcup_{i \in (L \cup U')} \bigwedge_{i \in I} t'_1 \models \Diamond \beta_{F(i)} \land \bigwedge_{i \in E_{L', U'} \bigcup F(i)} t_2 \models \Diamond \gamma_k \land \bigwedge_{i \in I \setminus (L \cup U')} t_2 \models \neg \Diamond \delta_k
\]

\[
7
\]
Here $F$ is a total function from $I - (I' \cup I'')$ to $\bigcup_{i \in I} N_i$ s.t. $\forall i. F(i) \in N_i$.

Putting it together again yields

$$
t_1 \models \Diamond (A^+ \land -A^- \land \neg\Diamond(-Act) \land \bigwedge_{i \in I} \Diamond\Psi_i \land \bigwedge_{j \in J} \neg\Diamond Y_j) \lor$$

$$\bigvee_{\alpha \in Act,I',I''} \left[ \bigwedge_{i \in I'} t_2 \models \neg\Diamond Y_j \land \bigwedge_{j \in J} t_2 \models \Diamond\Psi_i \land \bigwedge_{i \in I} \bigwedge_{k \in P(F(i))} t_2 \models \Diamond\gamma_k \right]$$

$$\bigwedge_{i \in I - (I' \cup I'')} t_2 \models \neg\Diamond\delta_k \land t_1 \models \Diamond (A^+ \cup \{ a \} \land -A^- \land$$

$$\bigwedge_{j \in J} \bigwedge_{k \in Act} \neg\Diamond (Y_j \land b) \land$$

$$\bigwedge_{i \in I''} \Diamond a_i \land \bigwedge_{i \in I - (I' \cup I'')} \Diamond F(i) \right]$$

This has the desired form. \hfill \Box

### 3.3 The Tableau-rules

Now we can define the rules for the construction of a tableau that decides $t \models \Diamond \Phi$ for $\Phi \in \mathcal{F}_d^\epsilon$. In this construction we assume that we can already decide all problems of the form $t' \models \Diamond \Psi$ or $t' \models \neg\Diamond \Psi$ for any $\Psi \in \mathcal{F}_{d-1}^\epsilon$. In the base case of $d = 0$ this condition is trivially satisfied, as $\mathcal{F}_0^\epsilon = \emptyset$. Also we assume that we can decide problems of the form $t \models \Diamond(-Act)$. (This is equivalent to $\exists t \not\rightarrow \epsilon$.

**Lemma 3.8** Let $t$ be a PA-term. It is decidable if $t \models \Diamond(-Act)$.

**Proof** The algorithm proceeds by successively marking variables as being reducible to $\epsilon$. First mark all variables $X$ s.t. $\exists X \not\rightarrow \epsilon$. Then mark all variables $Y$ s.t. $\exists X \not\rightarrow G$ where all variables occurring in $G$ are already marked. Repeat this until no new variables can be marked. Then $t \models \Diamond(-Act)$ iff all variables occurring in $t$ are marked. \hfill \Box

The nodes in the tableau are marked with sets of expressions of the form $t \models \Phi$, where $t$ is a PA-term and $\Phi$ an EF-formula. Such sets are denoted by $\Gamma$. These sets of expressions at the nodes are interpreted conjunctively, while the branches in the tableau are interpreted disjunctively. The tableau is successful iff there is a successful branch.

**PAR** \hspace{1cm} \hspace{1cm} $t_1 \parallel t_2 \models \Diamond \Phi$

\hspace{1cm} \hspace{1cm} see Lemma 3.6

**SEQ** \hspace{1cm} \hspace{1cm} $t_1 . t_2 \models \Diamond \Phi$

\hspace{1cm} \hspace{1cm} see Lemma 3.7
For all tableau-rules the antecedent is true iff one of the consequences is true.

Definition 3.10 (Termination conditions) A node \( n \) consisting of a set of formulae \( \Gamma \) is a terminal node if one of the following conditions is satisfied:

1. \( \Gamma \) is empty
2. \( t \vdash \Diamond \Psi \in \Gamma \) with \( \Psi \in \mathcal{F}_{d-1} \) and \( t \not\models \Diamond \Psi \)
3. \( t \vdash \neg \Diamond \Psi \in \Gamma \) with \( \Psi \in \mathcal{F}_{d-1} \) and \( t \models \Diamond \Psi \)
4. \( t \vdash \Diamond (\neg \text{Act}) \in \Gamma \) and \( \not\exists t \xrightarrow{a} \epsilon \)
5. \( t \vdash \neg \Diamond (\neg \text{Act}) \in \Gamma \) and \( \exists t \xrightarrow{a} \epsilon \)
6. \( t \vdash A^+ \in \Gamma \) and \( \exists a \in A^+. \not\exists t \xrightarrow{a} t' \)
7. \( t \vdash \neg A^- \in \Gamma \) and \( \exists a \in A^- . \exists t \rightarrow t' \)

8. There is a previous node \( n' \) in the same branch that is marked with set \( \Gamma' \) s.t. \( \Gamma = \Gamma' \)

Terminals of type 1 are successful, while terminals of type 2–8 are unsuccessful.

4 Soundness and Completeness

Lemma 4.1 If the root node has the form \( t \vdash \Diamond \Phi \), then for every node \( n \) in the tableau at least one of the following conditions is satisfied:

- A tableau rule is applicable
- The node is a terminal node.

Proof The only problematic cases are the formulae of the form \( t \vdash \neg \Diamond \Phi \). If such a formula occurs, then it must be due to the rules SEQ or Step. By definition of the rule Step and Lemma 3.7 we know that \( \Phi \in F_{d-1}^c \). Therefore the node is a terminal node or one of the rules Induct2 or Term2 is applicable.

Lemma 4.2 The tableau is finite.

Proof There are only finitely many formulae in \( F_d^c \) and only finitely many rules \( X \xrightarrow{\alpha} t \) with only finitely many subterms of the terms \( t \). So there are only finitely many different sets of expressions of the form \( t \vdash \Phi \) in the tableau. Therefore the branches of the tableau can only have finite length, because of termination condition 8. As the tableau is finitely branching the result follows.

Now we prove the soundness and completeness of the tableau.

Lemma 4.3 Let \( \Phi \in F_d^c \). If there is a successful tableau with root \( t \vdash \Diamond \Phi \), then \( t \models \Diamond \Phi \).

Proof A successful tableau has a successful branch ending with a node marked by the empty set of formulae. As these sets are interpreted conjunctively this node is true. By Proposition 3.9 all its ancestor-nodes must be true and thus the root-node must be true as well.

Lemma 4.4 Let \( t \) be a PA-term, \( \Phi \in F_d^c \) and \( \Gamma \) a set of formulae. If \( t \models \Diamond \Phi \) then there is a sequence of rule applications s.t. there is a path from a node marked \( \{ t \vdash \Diamond \Phi \} \cup \Gamma \) to a node marked \( \Gamma \).

Proof by induction on lexicographically ordered pairs \((x, y)\) where \( x \) is the length of the shortest sequence \( \sigma \) s.t. \( t \xrightarrow{\sigma} t' \) and \( t' \models \Phi \), and \( y \) is the size of \( t \).

The construction of the tableau is done in rounds. Each round consists of an application of one of the rules SEQ, PAR or Step, followed by several applications of the rules \( \wedge, \vee, \) Induct1, Induct2, Term1 and Term2 to clear away unnecessary formulae (Remember that these rules take precedence over the rules SEQ, PAR and Step). As the node is true at least one of its successors (at the end of the round) must be true.
SEQ If this rule was used, then the successor has the form $\Gamma \cup \Gamma'$, where all members of $\Gamma'$ are of the form $t' \vdash \square \Phi'$ where $t'$ is smaller than $t$. This means that $y$ is now smaller. An analysis of the proof of Lemma 3.7 shows that the value of $x$ cannot have increased. The result follows from the induction hypothesis.

PAR In this case the successor has the form $\{t_1 \vdash \square \Phi_1, t_2 \vdash \square \Phi_2\} \cup \Gamma$ s.t. $t_1$ and $t_2$ are smaller than $t$. It follows from the proof of Lemma 3.6 that the value of $x$ has not increased, while the value of $y$ is smaller. Applying the induction hypothesis twice yields the desired result.

Step Here we have two subcases:

1. If the first branch of the Step-rule is true, then applications of the rules $\land$, $\lor$, Induct1, Induct2, Term1, Term2, Act1 and Act2 directly lead to a node marked by $\Gamma$.

2. Otherwise choose the true successor that corresponds to the shortest sequence $\sigma$ (see above). Here the value of $y$ may have increased, but the value of $x$ has decreased by 1, and thus we can apply the induction hypothesis.

This construction cannot be stopped by termination condition 8, because this would contradict the minimality of the length of $\sigma$. \hfill ∎

Corollary 4.5 If $t \models \square \Phi$ for $\Phi \in F^c_\sigma$, then there is a successful tableau for $t \vdash \square \Phi$.

Proof Applying Lemma 4.4 for the special case of an empty set $\Gamma$ yields that a node can be reached that is marked by the empty set. The branch from the root-node to this node is successful and thus there is a successful tableau. \hfill ∎

Lemma 4.6 Let $t$ be a PA-term and $\Phi \in F^c_\sigma$. $t \models \square \Phi$ iff there is a successful tableau for $t \vdash \square \Phi$.

Proof Directly from Lemma 4.3 and Corollary 4.5. \hfill ∎

Theorem 4.7 The model checking problem for PA-processes and the logic EF is decidable.

Proof By Lemma 3.4 it suffices to prove decidability for formulae of the form $\square \Phi$ with $\Phi$ in $F^c_\sigma$ for any $d$. We prove this by induction on $d$. By Lemma 4.6 and Lemma 4.2 it suffices to construct a finite tableau. During the construction we need to decide problems of the form $t' \models \square \Psi$ for $\Psi \in F^c_{d-1}$ and problems of the form $t \models \square (\neg \text{Act})$. The first one is possible by induction hypothesis, and the second one by Lemma 3.8. \hfill ∎
5 Extensions

In this section we extend the logic $EF$ by constraints on sequences. So far the expression $t \models \Diamond \Phi$ only means that there is a sequence $\sigma$ s.t. $t \xrightarrow{\sigma} t'$ and $t' \models \Phi$ without saying anything about $\sigma$. Now we generalize the operator $\Diamond$ to $\Diamond C$, where $C : \text{Act}^* \mapsto \{\text{true}, \text{false}\}$ are predicates on finite sequences of actions. Here these functions are called constraints.

The semantics of the modified modal operator $\Diamond C$ is defined by:

$$\|\Diamond_C \Phi\| = \{t | \exists \sigma, t'. t \xrightarrow{\sigma} t' \land t' \models \|\Phi\| \land C(\sigma)\}$$

We'll show that for a special class of constraints $C$ the extended logic is still decidable for PA-processes.

**Definition 5.1 (Decomposable constraints)** Let $a \in \text{Act}$, $i, k \in \mathbb{N}$ and $\sigma$ a sequence of actions. Decomposable constraints are of the following form

$$C ::= W(\sigma) \geq i \mid W(\sigma) \leq i \mid [W(\sigma)]_k = i \mid C_1 \lor C_2 \mid C_1 \land C_2 \mid \text{first}(\sigma) = a$$

where $W : \text{Act}^* \mapsto \mathbb{N}$ is a function on sequences s.t. $W(\sigma_1 \sigma_2) = W(\sigma_1) + W(\sigma_2)$ for all $\sigma_1, \sigma_2$. (This implies that if $\sigma$ is the empty sequence, then $W(\sigma) = 0$).

These constraints are called “decomposable”, because a constraint $C$ on a sequence of actions $\sigma$ performed by a sequential- or parallel composition of processes $t_1$ and $t_2$ can be expressed by constraints on sequences performed by $t_1$ and $t_2$.

For example let $W$ be the function that counts the number of $a$-actions in a sequence. Now if $t_1 \models t_2 \xrightarrow{\sigma} t'_2$ and $[W(\sigma)]_3 = 0$ then there are sequences $\sigma_1, \sigma_2$ s.t. $t_1 \xrightarrow{\sigma_1} t'_1$ and $t_2 \xrightarrow{\sigma_2} t'_2$ and either $W(\sigma_1) = W(\sigma_2) = 0$ or $W(\sigma_1) = 1$ and $W(\sigma_2) = 2$ or $W(\sigma_1) = 2$ and $W(\sigma_2) = 1$.

**Definition 5.2** Let $EF_{DC}$ be the extension of $EF$ by modal operators $\Diamond C$, where $C$ is a decomposable constraint.

By using decomposability of the constraints the Lemmas 3.6 and 3.7 can be extended to the logic $EF_{DC}$. The tableau method can be adjusted accordingly and thus the logic $EF_{DC}$ is still decidable for PA-processes.

Let $\lambda$ be the empty sequence of actions. The modified tableau rules are:

- **PAR**
  $$t_1 \models t_2 \vdash \Diamond_C \Phi$$
  the modified Lemma 3.6

- **SEQ**
  $$t_1 \cdot t_2 \vdash \Diamond_C \Phi$$
  the modified Lemma 3.7

- **Split**
  $$\{t \vdash \Diamond C_1 \land C_2 \Phi\} \cup \Gamma$$
  $$\{t \vdash \Diamond C_1 \Phi\} \cup \Gamma$$

- **Clear**
  $$\{t \vdash \Diamond C_1 \land C_2 \Phi\} \cup \Gamma$$
  if $C_2$ is equal to $\text{true}$
In the rules Step1 and Step2 the new constraints $C_i$ are computed from the constraint $C$ and the action $a_i$ by

$$
Cons(C_1 \land C_2, a) := Cons(C_1, a) \land Cons(C_2, a)
$$

$$
Cons(W(\sigma) \geq i, a) := W(\sigma) \geq i - W(a)
$$

$$
Cons(W(\sigma) \leq i, a) := W(\sigma) \leq i - W(a)
$$

$$
Cons([W(\sigma)]_k = j, a) := [W(\sigma)]_k = [j - W(a)]_k
$$

$$
Cons(first(\sigma) = b, a) := if a = b then true else false
$$
terminal if $C$ is equal to $false$, i.e. $C = C' \land false$ or $C = C' \land W(\sigma) \leq k$ for some $k < 0$.

Note that only finitely many different constraints can occur in a tableau, because of the definition of the function $Cons$, the rule $Clear$ and this new termination condition. Thus the proofs of soundness and completeness of the tableau from section 3 carry over to the extended logic with constraints.

**Theorem 5.3** The model checking problem for PA-processes and the logic $EF_{DC}$ is decidable.

With decomposable constraints we can also express the usual one-step next operator by defining

$$[a] := \Diamond C$$

with $C := first(\sigma) = a \land length(\sigma) = 1$.

## 6 Conclusion

We have shown decidability of the model checking problem for the branching time temporal logic $EF$ and PA-processes. The exact complexity of the problem is left open. While for the special case of BPPs the problem is PSPACE-complete [May96a, May96b] the algorithm described here for PA has superexponential complexity.

It is interesting to compare the decidability results for branching time logics with the results for the linear time $\mu$-calculus. While model checking PA-processes with $EF$ is decidable, it is undecidable for the linear time $\mu$-calculus [BH96]. For Petri nets the situation is just the other way round. While model checking Petri nets with $EF$ is undecidable [Esp, Esp96], it is decidable for the linear time $\mu$-calculus [Esp]. This emphasizes the fact that PA-processes and Petri nets are incomparable models of concurrent systems. For the modal $\mu$-calculus the model checking problem is undecidable even for BPPs [Esp, Esp96].

<table>
<thead>
<tr>
<th></th>
<th>$EF$</th>
<th>linear time $\mu$-calc.</th>
<th>modal $\mu$-calc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Petri nets</td>
<td>undecidable</td>
<td>decidable, EXPSP.-hard</td>
<td>undecidable</td>
</tr>
<tr>
<td>PA</td>
<td><strong>decidable</strong></td>
<td>undecidable</td>
<td>undecidable</td>
</tr>
<tr>
<td>BPP</td>
<td>PSPACE-complete</td>
<td>decidable, EXPSP.-hard</td>
<td>undecidable</td>
</tr>
<tr>
<td>finite LTS</td>
<td>polynomial</td>
<td>PSPACE-complete</td>
<td>$\in$ NP $\cap$ co-NP</td>
</tr>
</tbody>
</table>

**References**


SFB 342: Methoden und Werkzeuge für die Nutzung paralleler Rechnerarchitekturen

bisher erschienen:

Reihe A

342/2/90 A Reinhard Fößmeier: Die Rolle der Lastverteilung bei der numerischen Parallelprogrammierung, Februar 1990
342/3/90 A Klaus-Jörn Lange, Peter Rossmanith: Two Results on Unambiguous Circuits, Februar 1990
342/5/90 A Reinhold Letz, Johann Schumann, Stephan Bayerl, Wolfgang Bibel: SETHEO: A High-Performance Theorem Prover
342/6/90 A Johann Schumann, Reinhold Letz: PARTHEO: A High Performance Parallel Theorem Prover
342/7/90 A Johann Schumann, Norbert Trapp, Martin van der Koelen: SETHEO/PARTHEO Users Manual
342/10/90 A Walter Vogler: Bisimulation and Action Refinement
342/11/90 A Jörg Desel, Javier Esparza: Reachability in Reversible Free-Choice Systems
342/12/90 A Rob van Glabbeek, Ursula Goltz: Equivalences and Refinement
<table>
<thead>
<tr>
<th>ISSN</th>
<th>Authors</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>342/13/90</td>
<td>Rob van Glabbeek</td>
<td>The Linear Time - Branching Time Spectrum</td>
</tr>
<tr>
<td>342/14/90</td>
<td>Johannes Bauer, Thomas Bemmerl, Thomas Treml</td>
<td>Leistungsanalyse von verteilten Beobachtungs- und Bewertungswerkzeugen</td>
</tr>
<tr>
<td>342/15/90</td>
<td>Peter Rossmanith</td>
<td>The Owner Concept for PRAMs</td>
</tr>
<tr>
<td>342/16/90</td>
<td>G. Böckle, S. Trosch</td>
<td>A Simulator for VLIW-Architectures</td>
</tr>
<tr>
<td>342/17/90</td>
<td>P. Slavkovsky, U. Rüde</td>
<td>Schnellere Berechnung klassischer Matrix-Multiplikationen</td>
</tr>
<tr>
<td>342/18/90</td>
<td>Christoph Zenger</td>
<td>SPARSE GRIDS</td>
</tr>
<tr>
<td>342/19/90</td>
<td>Michael Griebel, Michael Schneider, Christoph Zenger</td>
<td>A combination technique for the solution of sparse grid problems</td>
</tr>
<tr>
<td>342/20/90</td>
<td>Michael Griebel</td>
<td>A Parallelizable and Vectorizable Multi-Level-Algorithm on Sparse Grids</td>
</tr>
<tr>
<td>342/21/90</td>
<td>V. Diekert, E. Ochmanski, K. Reinhardt</td>
<td>On confluent semicommutations-decidability and complexity results</td>
</tr>
<tr>
<td>342/22/90</td>
<td>Manfred Broy, Claus Dendorfer</td>
<td>Functional Modelling of Operating System Structures by Timed Higher Order Stream Processing Functions</td>
</tr>
<tr>
<td>342/23/90</td>
<td>Rob van Glabbeek, Ursula Goltz</td>
<td>A Deadlock-sensitive Congruence for Action Refinement</td>
</tr>
<tr>
<td>342/24/90</td>
<td>Manfred Broy</td>
<td>On the Design and Verification of a Simple Distributed Spanning Tree Algorithm</td>
</tr>
<tr>
<td>342/27/90</td>
<td>Wolfgang Ertel</td>
<td>Random Competition: A Simple, but Efficient Method for Parallelizing Inference Systems</td>
</tr>
<tr>
<td>342/28/90</td>
<td>Rob van Glabbeek, Frits Vaandrager</td>
<td>Modular Specification of Process Algebras</td>
</tr>
<tr>
<td>342/29/90</td>
<td>Rob van Glabbeek, Peter Weijland</td>
<td>Branching Time and Abstraction in Bisimulation Semantics</td>
</tr>
<tr>
<td>Reihe A</td>
<td>Seite</td>
<td>Titel</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>342/30/90 A</td>
<td>Michael Griebel: Parallel Multigrid Methods on Sparse Grids</td>
<td></td>
</tr>
<tr>
<td>342/31/90 A</td>
<td>Rolf Niedermeier, Peter Rossmanith: Unambiguous Simulations of Auxiliary Pushdown Automata and Circuits</td>
<td></td>
</tr>
<tr>
<td>342/32/90 A</td>
<td>Inga Niepel, Peter Rossmanith: Uniform Circuits and Exclusive Read PRAMs</td>
<td></td>
</tr>
<tr>
<td>342/33/90 A</td>
<td>Dr. Hermann Hellwagner: A Survey of Virtually Shared Memory Schemes</td>
<td></td>
</tr>
<tr>
<td>342/1/91 A</td>
<td>Walter Vogler: Is Partial Order Semantics Necessary for Action Refinement?</td>
<td></td>
</tr>
<tr>
<td>342/2/91 A</td>
<td>Manfred Broy, Frank Dederichs, Claus Dendorfer, Rainer Weber: Characterizing the Behaviour of Reactive Systems by Trace Sets</td>
<td></td>
</tr>
<tr>
<td>342/3/91 A</td>
<td>Ulrich Furbach, Christian Suttner, Bertram Fronhöfer: Massively Parallel Inference Systems</td>
<td></td>
</tr>
<tr>
<td>342/4/91 A</td>
<td>Rudolf Bayer: Non-deterministic Computing, Transactions and Recursive Atomicity</td>
<td></td>
</tr>
<tr>
<td>342/5/91 A</td>
<td>Robert Gold: Dataflow semantics for Petri nets</td>
<td></td>
</tr>
<tr>
<td>342/6/91 A</td>
<td>A. Heise; C. Dimitrovici: Transformation und Komposition von P/T-Netzen unter Erhaltung wesentlicher Eigenschaften</td>
<td></td>
</tr>
<tr>
<td>342/7/91 A</td>
<td>Walter Vogler: Asynchronous Communication of Petri Nets and the Refinement of Transitions</td>
<td></td>
</tr>
<tr>
<td>342/8/91 A</td>
<td>Walter Vogler: Generalized OM-Bisimulation</td>
<td></td>
</tr>
<tr>
<td>342/9/91 A</td>
<td>Christoph Zenger, Klaus Hallatschek: Fouriertransformation auf dünnen Gittern mit hierarchischen Basen</td>
<td></td>
</tr>
<tr>
<td>342/10/91 A</td>
<td>Erwin Loibl, Hans Obermaier, Markus Pawlowski: Towards Parallelism in a Relational Database System</td>
<td></td>
</tr>
<tr>
<td>342/11/91 A</td>
<td>Michael Werner: Implementierung von Algorithmen zur Kompaktifizierung von Programmen für VLIW-Architekturen</td>
<td></td>
</tr>
<tr>
<td>342/12/91 A</td>
<td>Reiner Müller: Implementierung von Algorithmen zur Optimierung von Schleifen mit Hilfe von Software-Pipelining Techniken</td>
<td></td>
</tr>
<tr>
<td>342/13/91 A</td>
<td>Sally Baker, Hans-Jörg Beier, Thomas Bemmerl, Arndt Bode, Hubert Ertl, Udo Graf, Olav Hansen, Josef Haunerdinger, Paul Hofstetter, Rainer Knödlseder, Jaroslav Kremencik, Siegfried Langenbuch, Robert Lindhof, Thomas Ludwig, Peter Luksch, Roy Milner, Bernhard Ries, Thomas Treml: TOPSYS - Tools for Parallel Systems (Artikelsammlung); 2., erweiterte Auflage</td>
<td></td>
</tr>
</tbody>
</table>
342/14/91 A Michael Griebel: The combination technique for the sparse grid solution of PDE's on multiprocessor machines

342/15/91 A Thomas F. Gritzner, Manfred Broy: A Link Between Process Algebras and Abstract Relation Algebras?

342/16/91 A Thomas Bemmerl, Arndt Bode, Peter Braun, Olav Hansen, Thomas Treml, Roland Wismüller: The Design and Implementation of TOPSYS

342/17/91 A Ulrich Furbach: Answers for disjunctive logic programs

342/18/91 A Ulrich Furbach: Splitting as a source of parallelism in disjunctive logic programs

342/19/91 A Gerhard W. Zumbusch: Adaptive parallele Multilevel-Methoden zur Lösung elliptischer Randwertprobleme

342/20/91 A M. Jobmann, J. Schumann: Modelling and Performance Analysis of a Parallel Theorem Prover


342/22/91 A Wolfgang Ertel, Theodor Gemenis, Johann M. Ph. Schumann, Christian B. Suttner, Rainer Weber, Zongyan Qiu: Formalisms and Languages for Specifying Parallel Inference Systems

342/23/91 A Astrid Kiehn: Local and Global Causes

342/24/91 A Johann M.Ph. Schumann: Parallelization of Inference Systems by using an Abstract Machine

342/25/91 A Eike Jessen: Speedup Analysis by Hierarchical Load Decomposition


342/27/91 A Thomas Schnekenburger, Andreas Weiningerr, Michael Friedrich: Introduction to the Parallel and Distributed Programming Language ParMod-C

342/28/91 A Claus Dendorfer: Funktionale Modellierung eines Postsystems

342/29/91 A Michael Griebel: Multilevel algorithms considered as iterative methods on indefinite systems

342/30/91 A W. Reisig: Parallel Composition of Liveness

342/31/91 A Thomas Bemmerl, Christian Kasperbauer, Martin Mairandres, Bernhard Ries: Programming Tools for Distributed Multiprocessor Computing Environments
<table>
<thead>
<tr>
<th>Code</th>
<th>Author(s)</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>342/32/91 A</td>
<td>Frank Leßke</td>
<td>On constructive specifications of abstract data types using temporal logic</td>
</tr>
<tr>
<td>342/1/92 A</td>
<td>L. Kanal, C.B. Suttner (Editors)</td>
<td>Informal Proceedings of the Workshop on Parallel Processing for AI</td>
</tr>
<tr>
<td>342/2/92 A</td>
<td>Manfred Broy, Frank Dederichs, Claus Dendorfer, Max Fuchs, Thomas F. Gritzner, Rainer Weber</td>
<td>The Design of Distributed Systems - An Introduction to FOCUS</td>
</tr>
<tr>
<td>342/2-2/92 A</td>
<td>Manfred Broy, Frank Dederichs, Claus Dendorfer, Max Fuchs, Thomas F. Gritzner, Rainer Weber</td>
<td>The Design of Distributed Systems - An Introduction to FOCUS - Revised Version (erschienen im Januar 1993)</td>
</tr>
<tr>
<td>342/3/92 A</td>
<td>Manfred Broy, Frank Dederichs, Claus Dendorfer, Max Fuchs, Thomas F. Gritzner, Rainer Weber</td>
<td>Summary of Case Studies in FOCUS - a Design Method for Distributed Systems</td>
</tr>
<tr>
<td>342/5/92 A</td>
<td>Michael Friedrich</td>
<td>Sprachmittel und Werkzeuge zur Unterstützung paralleler und verteilter Programmierung</td>
</tr>
<tr>
<td>342/6/92 A</td>
<td>Thomas F. Gritzner</td>
<td>The Action Graph Model as a Link between Abstract Relation Algebras and Process-Algebraic Specifications</td>
</tr>
<tr>
<td>342/7/92 A</td>
<td>Sergei Gorlatch</td>
<td>Parallel Program Development for a Recursive Numerical Algorithm: a Case Study</td>
</tr>
<tr>
<td>342/8/92 A</td>
<td>Henning Spruth, Georg Sigl, Frank Johannes</td>
<td>Parallel Algorithms for Slicing Based Final Placement</td>
</tr>
<tr>
<td>342/9/92 A</td>
<td>Herbert Bauer, Christian Sporrer, Thomas Krodel</td>
<td>On Distributed Logic Simulation Using Time Warp</td>
</tr>
<tr>
<td>342/10/92 A</td>
<td>H. Bungartz, M. Griebel, U. Rüde</td>
<td>Extrapolation, Combination and Sparse Grid Techniques for Elliptic Boundary Value Problems</td>
</tr>
<tr>
<td>342/11/92 A</td>
<td>M. Griebel, W. Huber, U. Rüde, T. Störtkuhl</td>
<td>The Combination Technique for Parallel Sparse-Grid-Preconditioning and Solution of PDEs on Multiprocessor Machines and Workstation Networks</td>
</tr>
<tr>
<td>342/12/92 A</td>
<td>Rolf Niedermeier, Peter Rossmannith</td>
<td>Optimal Parallel Algorithms for Computing Recursively Defined Functions</td>
</tr>
<tr>
<td>342/13/92 A</td>
<td>Rainer Weber</td>
<td>Eine Methodik für die formale Anforderungsspezifikation verteilter Systeme</td>
</tr>
<tr>
<td>ID</td>
<td>Title</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>342/14/92 A</td>
<td>Michael Griebel: Grid- and point-oriented multilevel algorithms</td>
<td></td>
</tr>
<tr>
<td>342/15/92 A</td>
<td>M. Griebel, C. Zenger, S. Zimmer: Improved multilevel algorithms for full and sparse grid problems</td>
<td></td>
</tr>
<tr>
<td>342/16/92 A</td>
<td>J. Desel, D. Gomm, E. Kindler, B. Paech, R. Walter: Bausteine eines kompositionalen Beweiskalküls für netzmodellierte Systeme</td>
<td></td>
</tr>
<tr>
<td>342/17/92 A</td>
<td>Frank Dederichs: Transformation verteilter Systeme: Von applikativen zu prozeduralen Darstellungen</td>
<td></td>
</tr>
<tr>
<td>342/18/92 A</td>
<td>Andreas Listl, Markus Pawlowski: Parallel Cache Management of a RDBMS</td>
<td></td>
</tr>
<tr>
<td>342/19/92 A</td>
<td>Erwin Loibl, Markus Pawlowski, Christian Roth: PART: A Parallel Relational Toolbox as Basis for the Optimization and Interpretation of Parallel Queries</td>
<td></td>
</tr>
<tr>
<td>342/20/92 A</td>
<td>Jörg Desel, Wolfgang Reisig: The Synthesis Problem of Petri Nets</td>
<td></td>
</tr>
<tr>
<td>342/21/92 A</td>
<td>Robert Balder, Christoph Zenger: The d-dimensional Helmholtz equation on sparse Grids</td>
<td></td>
</tr>
<tr>
<td>342/22/92 A</td>
<td>Ilko Michler: Neuronale Netzwerk-Paradigmen zum Erlernen von Heuristiken</td>
<td></td>
</tr>
<tr>
<td>342/23/92 A</td>
<td>Wolfgang Reisig: Elements of a Temporal Logic. Coping with Concurrency</td>
<td></td>
</tr>
<tr>
<td>342/24/92 A</td>
<td>T. Störtkuhl, Chr. Zenger, S. Zimmer: An asymptotic solution for the singularity at the angular point of the lid driven cavity</td>
<td></td>
</tr>
<tr>
<td>342/25/92 A</td>
<td>Ekkart Kindler: Invariants, Compositionality and Substitution</td>
<td></td>
</tr>
<tr>
<td>342/26/92 A</td>
<td>Thomas Bonk, Ulrich Rüde: Performance Analysis and Optimization of Numerically Intensive Programs</td>
<td></td>
</tr>
<tr>
<td>342/1/93 A</td>
<td>M. Griebel, V. Thurner: The Efficient Solution of Fluid Dynamics Problems by the Combination Technique</td>
<td></td>
</tr>
<tr>
<td>342/2/93 A</td>
<td>Ketil Stølen, Frank Dederichs, Rainer Weber: Assumption / Commitment Rules for Networks of Asynchronously Communicating Agents</td>
<td></td>
</tr>
<tr>
<td>342/3/93 A</td>
<td>Thomas Schneekenburger: A Definition of Efficiency of Parallel Programs in Multi-Tasking Environments</td>
<td></td>
</tr>
</tbody>
</table>
Reihe A

342/5/93 A Manfred Kunde, Rolf Niedermeier, Peter Rossmanith: Faster Sorting and Routing on Grids with Diagonals

342/6/93 A Michael Griebel, Peter Oswald: Remarks on the Abstract Theory of Additive and Multiplicative Schwarz Algorithms

342/7/93 A Christian Sporrer, Herbert Bauer: Corolla Partitioning for Distributed Logic Simulation of VLSI Circuits

342/8/93 A Herbert Bauer, Christian Sporrer: Reducing Rollback Overhead in Time-Warp Based Distributed Simulation with Optimized Incremental State Saving

342/9/93 A Peter Slavkovsky: The Visibility Problem for Single-Valued Surface \( z = f(x,y) \): The Analysis and the Parallelization of Algorithms

342/10/93 A Ulrich Rüde: Multilevel, Extrapolation, and Sparse Grid Methods

342/11/93 A Hans Regler, Ulrich Rüde: Layout Optimization with Algebraic Multigrid Methods

342/12/93 A Dieter Barnard, Angelika Mader: Model Checking for the Modal Mu-Calculus using Gauß Elimination

342/13/93 A Christoph Pflaum, Ulrich Rüde: Gauß' Adaptive Relaxation for the Multilevel Solution of Partial Differential Equations on Sparse Grids

342/14/93 A Christoph Pflaum: Convergence of the Combination Technique for the Finite Element Solution of Poisson's Equation

342/15/93 A Michael Luby, Wolfgang Ertel: Optimal Parallelization of Las Vegas Algorithms

342/16/93 A Hans-Joachim Bungartz, Michael Griebel, Dierk Röschke, Christoph Zenger: Pointwise Convergence of the Combination Technique for Laplace's Equation

342/17/93 A Georg Stellner, Matthias Schumann, Stefan Lamberts, Thomas Ludwig, Arndt Bode, Martin Kiehl und Rainer Mehlhorn: Developing Multicomputer Applications on Networks of Workstations Using NXLib

342/18/93 A Max Fuchs, Ketil Stølen: Development of a Distributed Min/Max Component

342/19/93 A Johann K. Obermaier: Recovery and Transaction Management in Write-optimized Database Systems
<table>
<thead>
<tr>
<th>Conference Number</th>
<th>Title</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>342/20/93 A</td>
<td>Deriving Efficient Parallel Programs by Systematizing Coarsening Specification Parallelism</td>
<td>Sergej Gorlatc’h:</td>
</tr>
<tr>
<td>342/01/94 A</td>
<td>Parallel Adaptive Numerical Simulation</td>
<td>Reiner Hüttl, Michael Schneider:</td>
</tr>
<tr>
<td>342/02/94 A</td>
<td>Parallel Routing of VLSI Circuits Based on Net Independency</td>
<td>Henning Spruth, Frank Johannes:</td>
</tr>
<tr>
<td>342/03/94 A</td>
<td>Parallel Hierarchical Sea-of-Gates Router</td>
<td>Henning Spruth, Frank Johannes, Kurt Antreich: PHIroute:</td>
</tr>
<tr>
<td>342/04/94 A</td>
<td>Parallel Multiple Shooting for Optimal Control Problems Under NX/2</td>
<td>Martin Kiehl, Rainer Mehlhorn, Matthias Schumann:</td>
</tr>
<tr>
<td>342/05/94 A</td>
<td>Heuristic Optimization of Parallel Computations</td>
<td>Christian Suttner, Christoph Goller, Peter Krauss, Klaus-Jörn Lange,</td>
</tr>
<tr>
<td>342/06/94 A</td>
<td>Using Subpages for Cache Coherency Control in Parallel Database Systems</td>
<td>Andreas Listl:</td>
</tr>
<tr>
<td>342/07/94 A</td>
<td>Specification and Refinement of Finite Dataflow Networks - a Relational Approach</td>
<td>Manfred Broy, Ketil Stølen:</td>
</tr>
<tr>
<td>342/08/94 A</td>
<td>Funktionale Spezifikation eines Kommunikationsprotokolls</td>
<td>Katharina Spies:</td>
</tr>
<tr>
<td>342/09/94 A</td>
<td>Applying a New Search Space Partitioning Method to Parallel Test Generation for Sequential Circuits</td>
<td>Peter A. Krauss:</td>
</tr>
<tr>
<td>342/10/94 A</td>
<td>A Functional Rephrasing of the Assumption/Commitment Specification Style</td>
<td>Manfred Broy:</td>
</tr>
<tr>
<td>342/11/94 A</td>
<td>An Attempt to Embed a Restricted Version of SDL as a Target Language in Focus</td>
<td>Eckhardt Holz, Ketil Stølen:</td>
</tr>
<tr>
<td>342/12/94 A</td>
<td>A Multi-Level-Algorithm for the Finite-Element-Solution of General Second Order Elliptic Differential Equations on Adaptive Sparse Grids</td>
<td>Christoph Pflaum:</td>
</tr>
<tr>
<td>342/13/94 A</td>
<td>Summary of Case Studies in FOCUS - a Design Method for Distributed Systems</td>
<td>Manfred Broy, Max Fuchs, Thomas F. Gritzner, Bernhard Schätz, Katharina Spies, Ketil Stølen:</td>
</tr>
<tr>
<td>342/14/94 A</td>
<td>Technologieabhängigkeit von Spezifikationen digitaler Hardware</td>
<td>Maximilian Fuchs:</td>
</tr>
</tbody>
</table>
M. Griebel, P. Oswald: Tensor Product Type Subspace Splittings And Multilevel Iterative Methods For Anisotropic Problems

Gheorghe Ștefănescu: Algebra of Flownomials

Ketil Stølen: A Refinement Relation Supporting the Transition from Unbounded to Bounded Communication Buffers

Michael Griebel, Tilman Neuhoeffer: A Domain-Oriented Multilevel Algorithm-Implementation and Parallelization

Michael Griebel, Walter Huber: Turbulence Simulation on Sparse Grids Using the Combination Method

Johann Schumann: Using the Theorem Prover SETHEO for verifying the development of a Communication Protocol in FOCUS - A Case Study -

Hans-Joachim Bungartz: Higher Order Finite Elements on Sparse Grids

Tao Zhang, Seonglim Kang, Lester R. Lipsky: The Performance of Parallel Computers: Order Statistics and Amdahl's Law

Lester R. Lipsky, Appie van de Liefvoort: Transformation of the Kronecker Product of Identical Servers to a Reduced Product Space

Pierre Fiorini, Lester R. Lipsky, Wen-Jung Hsin, Appie van de Liefvoort: Auto-Correlation of Lag-k For Customers Departing From Semi-Markov Processes

Sascha Hilgenfeldt, Robert Balder, Christoph Zenger: Sparse Grids: Applications to Multi-dimensional Schrödinger Problems

Maximilian Fuchs: Formal Design of a Model-N Counter

Hans-Joachim Bungartz, Stefan Schulte: Coupled Problems in Microsystems Technology

Alexander Pfaffinger: Parallel Communication on Workstation Networks with Complex Topologies

Ketil Stølen: Assumption/Commitment Rules for Data-flow Networks - with an Emphasis on Completeness

Ketil Stølen, Max Fuchs: A Formal Method for Hardware/Software Co-Design

Thomas Schnekenburger: The ALDY Load Distribution System
342/12/95 A  Javier Esparza, Stefan Römer, Walter Vogler: An Improvement of McMillan’s Unfolding Algorithm

342/13/95 A  Stephan Melzer, Javier Esparza: Checking System Properties via Integer Programming

342/14/95 A  Radu Grosu, Ketil Stølen: A Denotational Model for Mobile Point-to-Point Dataflow Networks

342/15/95 A  Andrei Kovalyov, Javier Esparza: A Polynomial Algorithm to Compute the Concurrency Relation of Free-Choice Signal Transition Graphs

342/16/95 A  Bernhard Schätz, Katharina Spies: Formale Syntax zur logischen Kernsprache der Focus-Entwicklungsmethodik

342/17/95 A  Georg Stelner: Using CoCheck on a Network of Workstations

342/18/95 A  Arndt Bode, Thomas Ludwig, Vaidy Sunderam, Roland Wismüller: Workshop on PVM, MPI, Tools and Applications

342/19/95 A  Thomas Schneekenburger: Integration of Load Distribution into ParMod-C

342/20/95 A  Ketil Stølen: Refinement Principles Supporting the Transition from Asynchronous to Synchronous Communication

342/21/95 A  Andreas Listl, Giannis Bozas: Performance Gains Using Subpages for Cache Coherency Control

342/22/95 A  Volker Heun, Ernst W. Mayr: Embedding Graphs with Bounded Treewidth into Optimal Hypercubes

342/23/95 A  Petr Jančar, Javier Esparza: Deciding Finiteness of Petri Nets up to Bisimulation

342/24/95 A  M. Jung, U. Rüde: Implicit Extrapolation Methods for Variable Coefficient Problems

342/01/96 A  Michael Griebel, Tilman Neunhoeffer, Hans Regler: Algebraic Multigrid Methods for the Solution of the Navier-Stokes Equations in Complicated Geometries

342/02/96 A  Thomas Grauschopf, Michael Griebel, Hans Regler: Additive Multilevel-Preconditioners based on Bilinear Interpolation, Matrix Dependent Geometric Coarsening and Algebraic-Multigrid Coarsening for Second Order Elliptic PDEs

342/03/96 A  Volker Heun, Ernst W. Mayr: Optimal Dynamic Edge-Disjoint Embeddings of Complete Binary Trees into Hypercubes
Reihe A

342/04/96 A Thomas Huckle: Efficient Computation of Sparse Approximate Inverses

342/05/96 A Thomas Ludwig, Roland Wismüller, Vaidy Sunderam, Arndt Bode: OMIS — On-line Monitoring Interface Specification

342/06/96 A Ekkart Kindler: A Compositional Partial Order Semantics for Petri Net Components

342/07/96 A Richard Mayr: Some Results on Basic Parallel Processes

342/08/96 A Ralph Radermacher, Frank Weimer: INSEL Syntax-Bericht


342/10/96 A Stefan Lamberts, Thomas Ludwig, Christian Röder, Arndt Bode: PFSLib — A File System for Parallel Programming Environments

342/11/96 A Manfred Broy, Gheorghe Ștefănescu: The Algebra of Stream Processing Functions

342/12/96 A Javier Esparza: Reachability in Live and Safe Free-Choice Petri Nets is NP-complete

342/13/96 A Radu Grosu, Ketil Stølen: A Denotational Model for Mobile Many-to-Many Data-flow Networks

342/14/96 A Giannis Bozas, Michael Jaedicke, Andreas Listl, Bernhard Mitschang, Angelika Reiser, Stephan Zimmermann: On Transforming a Sequential SQL-DBMS into a Parallel One: First Results and Experiences of the MIDAS Project

342/15/96 A Richard Mayr: A Tableau System for Model Checking Petri Nets with a Fragment of the Linear Time μ-Calculus

342/16/96 A Ursula Hinkel, Katharina Spies: Anleitung zur Spezifikation von mobilen, dynamischen Focus-Netzen

342/17/96 A Richard Mayr: Model Checking PA-Processes
<table>
<thead>
<tr>
<th>SFB 342</th>
<th>Methoden und Werkzeuge für die Nutzung paralleler Rechnerarchitekturen</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reihe B</td>
<td></td>
</tr>
<tr>
<td>342/1/90 B</td>
<td>Wolfgang Reisig: Petri Nets and Algebraic Specifications</td>
</tr>
<tr>
<td>342/2/90 B</td>
<td>Jörg Desel: On Abstraction of Nets</td>
</tr>
<tr>
<td>342/3/90 B</td>
<td>Jörg Desel: Reduction and Design of Well-behaved Free-choice Systems</td>
</tr>
<tr>
<td>342/4/90 B</td>
<td>Franz Abstreiter, Michael Friedrich, Hans-Jürgen Plewan: Das Werkzeug runtime zur Beobachtung verteilter und paralleler Programme</td>
</tr>
<tr>
<td>342/1/91 B</td>
<td>Barbara Paechl: Concurrency as a Modality</td>
</tr>
<tr>
<td>342/2/91 B</td>
<td>Birgit Kandler, Markus Pawlowski: SAM: Eine Sortier-Toolbox - Anwenderbeschreibung</td>
</tr>
<tr>
<td>342/3/91 B</td>
<td>Erwin Loibl, Hans Obermaier, Markus Pawlowski: 2. Workshop über Parallelisierung von Datenbanksystemen</td>
</tr>
<tr>
<td>342/4/91 B</td>
<td>Werner Pohlmann: A Limitation of Distributed Simulation Methods</td>
</tr>
<tr>
<td>342/5/91 B</td>
<td>Dominik Gomm, Eckart Kindler: A Weakly Coherent Virtually Shared Memory Scheme: Formal Specification and Analysis</td>
</tr>
<tr>
<td>342/6/91 B</td>
<td>Dominik Gomm, Eckart Kindler: Causality Based Specification and Correctness Proof of a Virtually Shared Memory Scheme</td>
</tr>
<tr>
<td>342/7/91 B</td>
<td>W. Reisig: Concurrent Temporal Logic</td>
</tr>
<tr>
<td>342/1/92 B</td>
<td>Malte Grosse, Christian B. Suttner: A Parallel Algorithm for Set-of-Support</td>
</tr>
<tr>
<td></td>
<td>Christian B. Suttner: Parallel Computation of Multiple Sets-of-Support</td>
</tr>
<tr>
<td>342/2/92 B</td>
<td>Arndt Bode, Hartmut Wedekind: Parallelrechner: Theorie, Hardware, Software, Anwendungen</td>
</tr>
<tr>
<td>342/1/93 B</td>
<td>Max Fuchs: Funktionale Spezifikation einer Geschwindigkeitsregelung</td>
</tr>
<tr>
<td>342/2/93 B</td>
<td>Eckart Kindler: Sicherheits- und Lebendigkeitseigenschaften: Ein Literaturüberblick</td>
</tr>
<tr>
<td>342/1/94 B</td>
<td>Andreas Listl; Thomas Schneekenburger; Michael Friedrich: Zum Entwurf eines Prototypen für MIDAS</td>
</tr>
</tbody>
</table>