Nordic Journal of Computing 1(1994), 364–385.

LOCAL MODEL CHECKING FOR CONTEXT-FREE PROCESSES

HARDI HUNGAR
Computer Science Dept., University of Oldenburg
P.O. Box 2503, D-26111 Oldenburg, Germany

BERNHARD STEFFEN
Lehrstuhl für Programmiersysteme, Universität Passau
D-94030 Passau, Germany

Abstract. We present a local model checking algorithm that decides for a given context-free process whether it satisfies a property written in the alternation-free modal mu-calculus. Heart of this algorithm is a purely syntactical sound and complete formal system, which in contrast to the known tableaux techniques, uses intermediate second-order assertions. These assertions provide a finite representation of all the infinite state sets which may arise during the proof in terms of the finite representation of the context-free argument process. This is the key to the effectiveness of our local model checking procedure.

CR Classification: F.3.1, D.2.4

1. Introduction

Model checking provides a powerful tool for the automatic verification of behavioral systems. The corresponding standard algorithms fall into two classes: the iterative algorithms (cf. [14, 8, 12, 13]) and the tableaux-based algorithms (cf., e.g. [5, 4, 9, 18, 22, 23]). Whereas the former class usually yields higher efficiency in the worst case, the latter allows *local* model checking (cf. [22]), which avoid the investigation of parts of a process which are not relevant to the verification. Local model checking has been exploited by Bradfield and Stirling [5, 4] in order to construct a sound and complete tableau system for the full mu-calculus [17], which can deal with *infinite* transition systems. However, in this tableau system, a purely syntactical characterization of the validity of a formula cannot always be achieved. Thus their proof method is not effective in general.

In this paper we develop a local model checking algorithm that decides the alternation-free modal mu-calculus for *context-free* processes, i.e. for processes that are given in terms of a context-free grammar, or equivalently, as mutually recursive systems of finite state labelled transition systems. We adopt the second viewpoint in this paper, which directly leads to our notion of *procedural* transition systems. These process representations are

Received September 1993. Accepted September 1994.

standard finite-state labelled transition systems that are extended by introducing *recursive procedures* or, alternatively, *recursive action refinements*. The resulting processes may of course be infinite state.

For this class of processes, an *iterative* model checking algorithm has already been developed in [6]. The central idea behind that algorithm is to raise the standard iterative model checking techniques to *second order*: in contrast to the usual approaches, in which the set of formulae that are satisfied by a certain state are iteratively computed, this algorithm iteratively computes a *property transformer* for each state class of the finite process representation. These property transformers can then simply be applied to solve the model checking problem.

Here, we also exploit the idea of second-order reasoning. The heart of our model checking algorithm is a purely syntactic sound and complete formal system, which in contrast to the known tableau techniques (cf. e.g. [5, 4, 9, 18, 22]), uses intermediate second-order assertions. These assertions allow to deal compositionally with the sequential composition operator which arises implicitly in procedural transition systems: parts of the transition system, which belong to a particular procedure incarnation (expansion of a certain call transition), are sequentially composed with the part of the transition system representing the process behavior after the return from the called procedure. This is the key to the effectiveness of our local model checking procedure, because it allows the finite representation of all the infinite state sets which may arise during the proof in terms of the finite representation of the context-free argument process.

2. Processes and Formulae

In this section we introduce *process graphs* as the basic structure for modelling behavior, and more specifically, *context-free process systems* as finite representations of infinite process graphs, as well as the (alternation-free) modal mu calculus as a logic for specification.

2.1 Context-Free Process Systems

DEFINITION 1. ([16]) *A* process graph *is a quintuple* $G = \langle \mathcal{S}, Act, \rightarrow, s_0, s_e \rangle$ *where:*

- $\mathcal{S}$ *is a set of* states*;*
- Act *is a set of* actions*;*
- $\rightarrow \subseteq \mathcal{S} \times Act \times \mathcal{S}$ *is the* transition relation*; and*
- $s_0, s_e \in \mathcal{S}$ *are distinguished elements, the "start state" and the "end state".* s_0 *must be* originating *and* s_e *must be* terminating*. I.e. there are no* $a \in Act$ *and* $s' \in \mathcal{S}$ *with* $\langle s', a, s_0 \rangle \in \rightarrow$ *nor* $\langle s_e, a, s' \rangle \in \rightarrow$.

In the remainder of the paper we use $s \xrightarrow{a} s'$ in lieu of $\langle s, a, s' \rangle \in \rightarrow$, and we call s' an *a-derivative* of s. Finally a process graph is said to be *finite-state*, when $\mathcal{S}$ and Act are finite.

Intuitively, a process graph encodes the operational behavior of a process. The set $\mathcal{S}$ represents the set of states the process may enter, *Act* the set of actions the process may perform and $\rightarrow$ the state transitions that may result upon execution of the actions.

Process graphs will be inserted for transitions into other process graphs by identifying the start and end states of the inserted graph with the start and end state of the replaced transition. The conditions on start and end states ensure that the semantic effect of the insertion corresponds to the intuitive meaning of the execution of the inserted process at that point. Without these conditions, processes could be exited through the start state or entered through the end state.

As in [6], we represent *context-free processes*, which may have infinitely many states, by means of *context-free process systems*. A context-free process system is essentially a set of named finite process graphs whose set of actions contains the names of the system's process graphs. Transitions labelled with such a name are meant to represent the denoted process graph. Thus the possibly infinite process graph represented in this way will have multiple copies of the states of the named finite process graphs. Note the analogy to representations in terms of context-free grammars: the names of the process graphs correspond to the non-terminals and the atomic actions to the terminals of a context-free grammar.

An alternative way to interpret named transitions is to think of them as *procedure calls*, where each named process graph stands for a procedure declaration. We use the term procedural process graph for process graphs where some of the actions are actually names.

DEFINITION 2. *A* procedural process graph *(PPG) is a process graph where the set of actions is divided into two disjoint classes,* atomic actions *Act and* (action) names $\mathcal{N}$. *A procedural process graph is* guarded *if the start state has no P-derivative for any action name P.*

In analogy to Definition 1, a sextuple $\langle \Sigma_P, Act, \mathcal{N}, \rightarrow_P, \sigma_P, \epsilon_P \rangle$ will denote a procedural process graph. We will use the term "state class" for the elements of Σ_P, and they will be denoted by lowercase greek letters.

In the following we shall only consider guarded procedural process graphs. This guarantees that the process graph represented by a context-free process system is finitely branching. The reason for indexing constituents of a procedural process graph will become apparent in the next definition.

DEFINITION 3. *A* context-free process system *(CFPS) is defined as a sextuple $\mathcal{P} = \langle Act, \mathcal{N}, \Delta, P_0, s_0, s_e \rangle$, where*

- *Act is a set of* actions*;*
- $\mathcal{N} = \{P_0, \ldots, P_{n-1}\}$ *is a set of* names*;*
- $\Delta =_{\mathrm{df}} \{ P_i = G_i \mid 0 \leq i < n \}$ *is a finite set of* PPG definitions *where the G_i are finite PPGs with names in $\mathcal{N}$ and atomic actions in Act*
- P_0 *is the* "main" PPG*, and*
- s_0 *and* s_e *are the start and the end state of the system.*

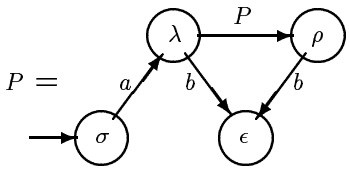


Fig. 1: A context-free process system for $a^n b^n$.

A CFPS $\mathcal{P}$ serves as a finite representation of the *complete expansion* of P_0, which we define below. Representable processes are exactly the BPA processes of [2], which form a subclass of the pushdown transitions graphs of [10]. In addition we define a *faithful* expansion, which is semantically equivalent, but contains some additional, semantically irrelevant states. These states are important for our proof system, as they allow us to get a handle on the effect of the component PPGs defining the CFPS under consideration.

DEFINITION 4. *Let P one of the names in a CFPS $\mathcal{P}$. The* complete expansion *of P wrt. $\mathcal{P}$ results from the process graph $s_0 \xrightarrow{P} s_e$ by successively replacing each transition $s \xrightarrow{P_i} s'$ by a copy of the corresponding PPG G_i, while identifying s with the start state of the copy and s' with its end state. Thus, a new state for each state class except the start and end state class is introduced. We denote the complete expansion of P by $Exp_{\mathcal{P}}(P)$.*

$FExp_{\mathcal{P}}(P)$ is the faithful complete expansion *of P. Its construction additionally introduces a copy of the start state class of the called PPG when expanding a named transition $s \xrightarrow{P_i} s'$. This copy has all the derivatives s has within the new copy of G_i, but none outside of it, and it is itself not reachable from any state.*

$Exp_{\mathcal{P}}(P)$ and $FExp_{\mathcal{P}}(P)$ are of course unique up to isomorphism, thus well defined. While considering the complete expansion as the 'real' semantics of a CFPS, our formal arguments will be based on the faithful expansion.

The PPG of Fig. 1 gives a CFPS which can not be replaced by an equivalent finite process graph. Its language is the set $\{\, a^n b^n \mid n \geq 1 \,\}$. Fig. 2 illustrates the system's stepwise expansion.

$Exp_{\mathcal{P}}(P_0)$ has a regular structure. At any point where during the expansion one transition named with P was expanded, a subgraph is *embedded*, which is isomorphic to $Exp_{\mathcal{P}}(P)$. The next definition describes this embedding operation. An even more restricted embedding is achieved in the case of a *faithful* expansion.

An embedding of a PG G in a PG G' requires that G is a subgraph of G'. No edges between internal nodes of G must be added. But there might be additional edges terminating at arbitrary nodes of G or originating at its

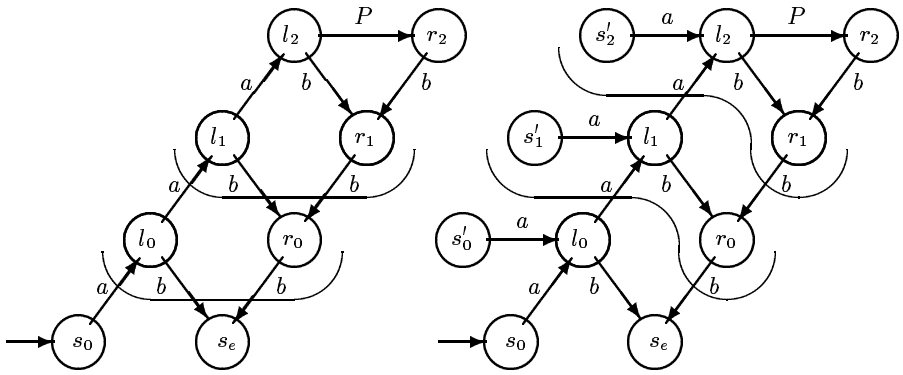


Fig. 2: The expansion and faithful expansion of P after the third expansion step. The thin lines on the left mark the borders of embeddings of $Exp_{\mathcal{P}}(P)$ in itself. These are achieved by mapping each state to the first, resp. second, state diagonally above it. Similarily, $Exp_{\mathcal{P}}(P)$ can be faithfully embedded in $FExp_{\mathcal{P}}(P)$. The mapping is done analogously, only that the start state is mapped to the appropriate s'_i.

start or end state. The embedding is faithful, if only the end state and not the start state serves as an exit.

DEFINITION 5. *An* embedding *of a process graph* $G = \langle \mathcal{S}, Act, \rightarrow, s_0, s_e \rangle$ *into another process graph* $G' = \langle \mathcal{S}', Act, \rightarrow', s'_0, s'_e \rangle$ *is an injection* ι *from* $\mathcal{S}$ *to* $\mathcal{S}'$ *satisfying*

- $s \xrightarrow{a} t$ *implies* $\iota(s) \xrightarrow{a}{}' \iota(t)$,
- $\iota(s) \xrightarrow{a}{}' \iota(t)$ *implies* $s \xrightarrow{a} t$ *or* $s, t \in \{s_0, s_e\}$, *and*
- *for* $s' \notin \iota(\mathcal{S})$, $\iota(t) \xrightarrow{a}{}' s'$ *implies* $t \in \{s_0, s_e\}$.

The embedding is faithful *if only* $\iota(s_e) \xrightarrow{a}{}' s'$ *is permitted in the third condition.*

As will be seen in Section 2.3, faithfulness is needed in the definition of second-order validity for start states. All the other states are not affected.

Conventions: We assume that the PPGs of a CFPS have mutually disjoint sets of state classes. The union of the sets of state classes of the PPGs of a CFPS $\mathcal{P}$ together with its start and end state form the set of *state classes of* $\mathcal{P}$. We write $\sigma \in P$ if σ is one of the state classes of the process graph named by P. Each state $s \in FExp_{\mathcal{P}}(P_0)$ can be assigned a unique state class of which it is a copy. Let this state class be denoted by $[s]$. Note that $[s]$ can not be an end state class of one of $\mathcal{P}$'s PPGs, since no copies of these classes are made during the expansion. In $Exp_{\mathcal{P}}(P_0)$, also start state classes of a PPG are empty. We define $[s_0] =_{\mathrm{df}} s_0$ and $[s_e] =_{\mathrm{df}} s_e$, and we write $s \in \sigma$, if $\sigma = [s]$ or if $\sigma = \epsilon_P$ and ϵ_P is identified with s in some expansion step.

$end(s)$ will denote the return state of the expansion step which created s, i.e. if s is one of the states of the copy of G_i which expanded $s' \xrightarrow{P_i} s''$, then $end(s) = s''$. Note that $end(s)$ does not belong to the same procedure incarnation as s, but to a surrounding one. $end(s_0)$ and $end(s_e)$ are defined to mean s_e.

Remember that $FExpPG(P_i)$ starts by adding a copy of all state classes of G_i except its end state class to $s_0 \xrightarrow{P_i} s_e$. Let this copy of σ for $\sigma \in G_i - \{\epsilon_{P_i}\}$ be denoted by $\mathit{first\_copy}(\sigma)$. And we choose $\mathit{first\_copy}(\epsilon_{P_i})$ to stand for s_e.

2.2 Mu Calculus

The following negation–free syntax defines a sublanguage of the mu-calculus, which in spite of being as expressive as the full mu-calculus allows a simpler technical development.

$$\phi ::= \mathit{ff} \mid tt \mid X \mid \phi \wedge \phi \mid \phi \vee \phi \mid [a]\phi \mid \langle a\rangle\phi \mid \nu X.\,\phi \mid \mu X.\,\phi$$

In the above, $a \in Act$, and $X \in Var$, where Var is a set of variables. The fixpoint operators νX and μX bind the occurrences of X in the formula behind the dot in the usual sense. Properties will be specified by *closed* formulae, i.e. formulae that do not contain any free variable. A formula is *alternation free* if no ν-subformula has a free variable which, in the context of the whole formula, is bound by a μ, and vice versa. The set of closed alternation-free formulae is denoted by $\mathcal{F}$.

There are no atoms in this calculus other than tt and ff, in order to simplify the presentation. It is, however, straightforward to add further constants, as long as the corresponding valuations respect the partitioning into state classes.

Formulae are interpreted with respect to a fixed (possibly infinite) process graph $G = \langle \mathcal{S}, Act, \rightarrow, s_0, s_e\rangle$, and an environment $e : Var \rightarrow 2^{\mathcal{S}}$.

$$\begin{aligned}
[\![\mathit{ff}]\!]e &= \emptyset \\
[\![tt]\!]e &= \mathcal{S} \\
[\![\phi_1 \vee \phi_2]\!]e &= [\![\phi_1]\!]e \cup [\![\phi_2]\!]e \\
[\![\phi_1 \wedge \phi_2]\!]e &= [\![\phi_1]\!]e \cap [\![\phi_2]\!]e \\
[\![[a]\phi]\!]e &= \{\, s \mid \forall s'.\ s \xrightarrow{a} s' \Rightarrow s' \in [\![\phi]\!]e \,\} \\
[\![\langle a\rangle\phi]\!]e &= \{\, s \mid \exists s'.\ s \xrightarrow{a} s' \wedge s' \in [\![\phi]\!]e \,\} \\
[\![\nu X.\phi]\!]e &= \bigcup\{S' \subseteq \mathcal{S} \mid S' \subseteq [\![\phi]\!]e[X \mapsto S']\} \\
[\![\mu X.\phi]\!]e &= \bigcap\{S' \subseteq \mathcal{S} \mid S' \supseteq [\![\phi]\!]e[X \mapsto S']\}
\end{aligned}$$

Intuitively, the semantic function maps a formula (with free variables) to the set of states for which the formula is "true". Accordingly, a state s satisfies X if s is an element of the set bound to X in e.

Since $[\![\phi]\!]e$ does not depend on e for a closed ϕ, we can write $s \models \phi$ instead of $s \in [\![\phi]\!]e$ for all e. We will use indices ($\models_G$, $[\![\phi]\!]_G$) if the process graph G is not clear from the context. $G \models \phi$ stands for $s_0 \models \phi$, $\mathcal{P} \models \phi$ for $s_0 \models_{Exp_{\mathcal{P}}(P_0)} \phi$ (a process graph satisfies a formula if its start state satisfies the formula, and a formula is valid in a context-free process system if it is valid in its complete expansion). Finally, $s \models \Phi$ for a set of formulae Φ abbreviates $s \models \phi$ for all members ϕ of Φ.

One way to examine the meaning of a formula is by inspecting its immediate subformulae, while repeatedly *unfolding* fixpoints. This procedure is formalized in the rules for the construction of a tableau. The set of formulae generated in this way is called the *closure* of the formula.

DEFINITION 6. *The* closure $CL(\phi)$ *of a (closed) formula* ϕ *is inductively defined as follows.*

$$\begin{aligned}
CL(tt) &= \emptyset \\
CL(ff) &= \emptyset \\
CL(\phi_1 \vee \phi_2) &= \{\, \phi_1 \vee \phi_2 \,\} \cup CL(\phi_1) \cup CL(\phi_2) \\
CL(\phi_1 \wedge \phi_2) &= \{\, \phi_1 \wedge \phi_2 \,\} \cup CL(\phi_1) \cup CL(\phi_2) \\
CL(\langle a \rangle \phi) &= \{\, \langle a \rangle \phi \,\} \cup CL(\phi) \\
CL([a]\phi) &= \{\, [a]\phi \,\} \cup CL(\phi) \\
CL(\nu X.\phi) &= \{\, \nu X.\phi \,\} \cup CL(\phi[\nu X.\phi/X]) \\
CL(\mu X.\phi) &= \{\, \mu X.\phi \,\} \cup CL(\phi[\mu X.\phi/X])
\end{aligned}$$

Note that, although a fixpoint may be unfolded infinitely often, the closure of a formula (which is different from the set of subformulae, due to the unfoldings of fixpoints) is always finite.

2.3 Second-Order Semantics

The validity of a formula is defined with respect to single states. To define the validity of a formula for a state class does not make much sense: the truth value might not be the same for different representatives (copies) of the same state class. But if two states s and s' belong to the same state class and the corresponding end states $end(s)$ and $end(s')$ satisfy the same set of formulae, then so do s and s'. This was the key observation motivating the *second-order semantics* in [6]. Here, the concept is captured by the notion of a second-order assertion. It is an analog to a pre/post specification in the style of Hoare's logic: the second part of a second-order assertion are the properties supposed to hold at the end state (the postcondition), and the first part gives a property holding at the given state, if the postcondition is satisfied.

DEFINITION 7. *A (closed)* second-order assertion *is a pair* $\langle \phi, \Theta \rangle$ *where* $\phi \in \mathcal{F}$ *and* $\Theta \subseteq \mathcal{F}$.
If s *is a state of a PG* G, $s \models_G \langle \phi, \Theta \rangle$ *iff* $\iota(s) \models \phi$ *for all faithful embeddings* ι *of* G *where* $\iota(s_e) \models \Theta$.
A PG G *satisfies a second-order assertion if its start state satisfies the assertion.*

In order to give meaning to assertions about the start state of a PPG within the complete expansion of a CFPS via an embedding one must guarantee that the representing node does not have additional (spoiling) transitions. This requirement is met by faithful embeddings. They allow us to separately deal with the behavior coming from the considered PPG and the (context-dependent) effect of the other transitions. This separation is the key for allowing an inductive reasoning. Except for this technical difference, the two versions of expansion are equivalent, because they have the same set of reachable states. In particular they satisfy the same set of formulae.

The usefulness of second-order assertions relies on the following fact.

FACT 1. If ι faithfully embeds G in G' and $\iota(s) \models_{G'} \phi$ for some $\phi \in \mathcal{F}$ and some state s of G, then $s \models_G \langle \phi, \{ \psi \in CL(\phi) \mid \iota(s_e) \in [\![\psi]\!]_{G'} \} \rangle$.

An intuitive explanation for the validity of this statement is given by the observation that to determine the truth value of a mu-calculus formula, it is sufficient to know the truth values of all closure formulae at future states. Since every path exiting $\iota(G)$ goes through $\iota(s_e)$, it is sufficient to have complete information about this cut point.

Since the identity faithfully embeds any PG G in itself and the end state is a deadlocked state, we immediately obtain:

G satisfies ϕ iff it satisfies $\langle \phi, \Theta_{dead} \rangle$,

where Θ_{dead} is the set of formulae true at a deadlocked state.

To reason compositionally about the validity of formulae on the level of the syntactic representation of a PG in terms of a CFPS, we introduce second-order sequents and define their validity.

DEFINITION 8. *A* second-order sequent *is a pair consisting of a state class of a CFPS and a second-order assertion, and it is written in the form* $\sigma \vdash \langle \phi, \Theta \rangle$, *where* Θ *has to be finite. The sequent* $\sigma \vdash \langle \phi, \Theta \rangle$ *is* valid, *iff* $first\_copy(\sigma) \models_{FExp_{\mathcal{P}}(P)} \langle \phi, \Theta \rangle$ *for* $\sigma \in P$, *and it is* exact, *iff*

- $\sigma \neq \epsilon_P$ *and in* $FExp_{\mathcal{P}}(P_0)$ *there is some* $s \in \sigma \cap [\![\phi]\!]$ *where* $\Theta = \{ \theta \in CL(\phi) \mid end(s) \models \theta \}$, *or*
- $\sigma = \epsilon_P$ *and in* $FExp_{\mathcal{P}}(P_0)$ *there is some* $s \in \sigma \cap [\![\phi]\!]$ *where* $\Theta = \{ \theta \in CL(\phi) \mid s \models \theta \}$.

A consequence of this definition is that if $\sigma \vdash \langle \phi, \Theta \rangle$ is valid, then all $s \in \sigma$ with $end(s) \models \Theta$ satisfy ϕ. Exact are those sequents where the second component is a set of formulae which actually occurs as validity set of some end state. Fact 1 tells us that every exact sequent is valid.

3. The Tableau System

One way to characterize the difference between regular and context free processes is the generalization from action prefixing to the usual sequential composition, which is equivalent to the generalization from left recursion to general (parameterless) recursion in this setting. In fact, the main problem of the construction of a tableaux system for context-free processes, which directly works on a CFPS representation, is to compositionally deal with the sequential composition, which implicitly arises when dealing with CFPS: parts of the transition system, which belong to a particular procedure incarnation (expansion of a certain call transition), are sequentially composed with the part of the transition system representing the process behaviour after the return from the called procedure. The solution to this problem are the modality rules, which like the composition rule of the Hoare Calculus, require the right guess of the intermediate formula. The rest of the rules are adaptations of the usual tableau rules (cf. eg. [18, 22, 9, 5, 4]), which organize the verification of the intermediate formulae. Formally our tableau proof rules are given below:

Start Rule

$$\frac{s_0 \vdash \phi}{\sigma_{P_0} \vdash \langle \phi, \Theta \rangle \quad \{ s_e \vdash \theta \mid \theta \in \Theta \}}$$

End Rules

$$\frac{s_e \vdash \phi \wedge \psi}{s_e \vdash \phi \quad s_e \vdash \psi}$$

$$\frac{s_e \vdash \phi \vee \psi}{s_e \vdash \phi} \qquad \frac{s_e \vdash \phi \vee \psi}{s_e \vdash \psi}$$

$$\frac{s_e \vdash \nu X.\phi}{s_e \vdash \phi[tt/X]} \qquad \frac{s_e \vdash \mu X.\phi}{s_e \vdash \phi[ff/X]}$$

Conjunction and Disjunction Rules

$$\frac{\sigma \vdash \langle \phi \wedge \psi, \Theta \rangle}{\sigma \vdash \langle \phi, \Theta \rangle \quad \sigma \vdash \langle \psi, \Theta \rangle}$$

$$\frac{\sigma \vdash \langle \phi \vee \psi, \Theta \rangle}{\sigma \vdash \langle \phi, \Theta \rangle} \qquad \frac{\sigma \vdash \langle \phi \vee \psi, \Theta \rangle}{\sigma \vdash \langle \psi, \Theta \rangle}$$

Modality Rules

$$\frac{\sigma \vdash \langle [a]\phi,\, \Theta\rangle}{\sigma' \vdash \langle \phi,\, \Theta\rangle \quad \ldots \quad \sigma_P \vdash \langle [a]\phi,\, \Psi\rangle \;\; \{\, \sigma'' \vdash \langle \psi,\, \Theta\rangle \mid \psi \in \Psi \,\}}$$

(all σ' where $\sigma \xrightarrow{a} \sigma'$ and all P, σ'' where $\sigma \xrightarrow{P} \sigma''$)

$$\frac{\sigma \vdash \langle \langle a\rangle\phi,\, \Theta\rangle}{\sigma' \vdash \langle \phi,\, \Theta\rangle} \qquad \frac{\sigma \vdash \langle \langle a\rangle\phi,\, \Theta\rangle}{\sigma_P \vdash \langle \langle a\rangle\phi,\, \Psi\rangle \quad \{\, \sigma'' \vdash \langle \psi,\, \Theta\rangle \mid \psi \in \Psi \,\}}$$

$(\sigma \xrightarrow{a} \sigma')$ $\qquad$ $(\sigma \xrightarrow{P} \sigma'')$

Fixpoint Rules

$$\frac{\sigma \vdash \langle \nu X.\phi,\, \Theta\rangle}{\sigma \vdash \langle \phi[\nu X.\phi/X],\, \Theta\rangle} \qquad \frac{\sigma \vdash \langle \mu X.\phi,\, \Theta\rangle}{\sigma \vdash \langle \phi[\mu X.\phi/X],\, \Theta\rangle}$$

Weakening Rule

$$\frac{\sigma \vdash \langle \phi,\, \Theta'\rangle}{\sigma \vdash \langle \phi,\, \Theta\rangle}\ \Theta \subset \Theta'$$

The Start Rule switches from a formula ϕ to a second-order assertion $\langle \phi,\, \Theta\rangle$. The formulae in Θ have to be verified for the global end state s_e with the End Rules. These five rules constitute a subsystem suited for the derivation of formulae valid at a deadlocked state. All the other rules are required for the verification of $\sigma \vdash \langle \phi,\, \Theta\rangle$. Here, the Conjunction and Disjunction Rules are obvious, as well as the Weakening Rule. The Fixpoint Rules do the usual unfolding. The remaining Modality Rules are the heart of the calculus. On their standard action transition part they are straightforward adaptations of the rules from usual tableau systems, but for call transitions they reflect the implicit sequential composition mentioned above: they split the proof into two parts, the part within the called procedure and the part after the return.

Thus the intuition behind the two kinds of Modality Rules is the same. The Diamond Rules only appear to be simpler as the nature of $\langle\ \rangle$ only requires the success along one branch, which leads to two relatively simple rules, whereas the Box Rule must collect proof obligations for each branch of the system requiring one complex rule. We will discuss the Diamond Rules in more details and leave the 'technical collapse' leading to the Box Rule to the reader. The first Diamond Rule is the straightforward adaptation for the standard action transition part, and the second rule deals with call transitions as follows: in order to proof the sequent $\sigma \vdash \langle \langle a\rangle\phi,\, \Theta\rangle$ one may first establish that the called PPG satisfies $\sigma_P \vdash \langle \langle a\rangle\phi,\, \Psi\rangle$ for some 'postcondition' Ψ. Of course, in order to complete the proof, this 'postcondition' Ψ must be verified for the current proof context leading to the proof obligations

$\sigma'' \vdash \langle \psi, \Theta \rangle$ for $\psi \in \Psi$, which are abbreviated by $\{ \sigma'' \vdash \langle \psi, \Theta \rangle \mid \psi \in \Psi \}$ in our tableaux.

A tableau built according to the rules is a partial proof for the sequent at its root. It counts as a complete proof if it is *successful*:

DEFINITION 9. (SUCCESSFUL TABLEAUX) *A finite tableau built according to the rules of this system is* successful *if every of its leaves is successful. A leaf n is* successful *if either*

- $n = s_e \vdash tt$
- $n = \sigma \vdash \langle tt, \Theta \rangle$
- $n = s_e \vdash [a]\phi$
- $n = \sigma \vdash \langle [a]\phi, \Theta \rangle$, *where σ is no end node of a PPG and there is no σ' with $\sigma \xrightarrow{a} \sigma'$ or $\sigma \xrightarrow{P} \sigma'$*
- $n = \epsilon_P \vdash \langle \phi, \Theta \rangle$ *and $\phi \in \Theta$, or*
- $n = \sigma \vdash \langle \phi(\nu X.\psi), \Theta \rangle$, *where $\phi(\nu X.\psi) \in CL(\nu X.\psi)$ and there is a node on the path from the root of the tableau to n labelled with the same sequent.*

A sequent is derivable *if there is a successful tableau with the sequent at its root. A formula ϕ is derivable for a CFPS $\mathcal{P}$ with start state s_0 if the sequent $s_0 \vdash \phi$ is derivable.*

Whereas the correctness of the first five conditions is rather obvious, the last one requires some explanation. Usually, one would allow to stop the process of expanding a tableau if a sequent $\sigma \vdash \langle \nu X.\psi, \Theta \rangle$ recurs. This is based on the argument that if, by unfolding the fixpoint formula once, no state $s \in \sigma$ with $end(s) \in [\![\Theta]\!]$ violating the fixpoint formula is found, then also by further unfoldings no such states will be found. So all of them satisfy the maximal fixpoint formula. Here, the situation is slightly more complex. It is true that whenever a recurrence according to the condition above occurs, a maximal fixpoint subformula $\nu X.\psi$ must have been unfolded. But no sequent of the form $\sigma \vdash \langle \nu X.\psi, \Theta \rangle$ need to recur. Our generous ν-success condition takes care of this phenomenon by allowing to stop as soon as an element of the closure of a ν-formula recurs.

An example of a successful tableau is given in Fig. 3, where M abbreviates $\mu Y.[b]Y$ and N stands for $\nu X.([a,b]X \wedge M)$. It proves that N holds for the CFPS of Fig. 1, i.e. that no infinite b-sequences are possible.

In this tableau, the simple version of ν-recurrence was sufficient. But our second example tableau, presented in Fig. 4, demonstrates the necessity of a more generous criterion. The sequent $\nu \vdash \langle \nu X.\,[a][b]X,\ \{[b]\nu X.\,[a][b]X\} \rangle$ would appear infinitely often in a tableau, but never twice on a path. The occurences would all be in finite subtableaux on sidebranches of the infinite derivation path. But the sequent marked with $\bullet$ recurs, and it contains a formula of the closure of $\nu X.\,[a][b]X$.

So we have to allow ν-recurrence in a more general form. But in contrast to [4, 5], our system does not require an explicit μ-success, although we

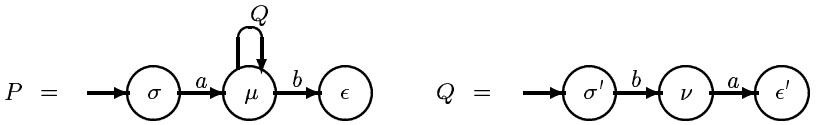
$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{P \vdash N}}{T_{e\,M} \quad T_{e\,N} \quad \sigma \vdash \langle N, \{M,N\}\rangle}}{\sigma \vdash \langle [a,b]N \wedge M, \{M,N\}\rangle}}{\sigma \vdash \langle [a,b]N, \{M,N\}\rangle \quad \sigma \vdash \langle M, \{M,N\}\rangle}}{\bullet\ \lambda \vdash \langle N, \{M,N\}\rangle \quad \sigma \vdash \langle [b]M, \{M,N\}\rangle}}{\lambda \vdash \langle [a,b]N \wedge M, \{M,N\}\rangle}}{\lambda \vdash \langle [a,b]N, \{M,N\}\rangle \quad T_\lambda}}{\sigma \vdash \langle [a,b]N, \{M,N\}\rangle \quad T_{\rho\,M} \quad T_{\rho\,N} \quad \epsilon \vdash \langle N, \{M,N\}\rangle}}{\lambda \vdash \langle N, \{M,N\}\rangle \hookrightarrow \bullet}$$

$$T_{e\,M} = \dfrac{s_e \vdash M}{s_e \vdash [b]ff} \qquad T_{e\,N} = \dfrac{\dfrac{s_e \vdash N}{s_e \vdash [a,b]tt \wedge M}}{s_e \vdash [a,b]tt \quad T_{e\,M}}$$

$$T_{\rho\,M} = \dfrac{\dfrac{\rho \vdash \langle M, \{M,N\}\rangle}{\rho \vdash \langle [b]M, \{M,N\}\rangle}}{\epsilon \vdash \langle M, \{M,N\}\rangle} \qquad T_{\rho\,N} = \dfrac{\dfrac{\dfrac{\rho \vdash \langle N, \{M,N\}\rangle}{\rho \vdash \langle [a,b]N \wedge M, \{M,N\}\rangle}}{\rho \vdash \langle [a,b]N, \{M,N\}\rangle} \quad T_{\rho\,M}}{\epsilon \vdash \langle N, \{M,N\}\rangle}$$

$$T_\lambda = \dfrac{\dfrac{\lambda \vdash \langle M, \{M,N\}\rangle}{\lambda \vdash \langle [b]M, \{M,N\}\rangle}}{\epsilon \vdash \langle M, \{M,N\}\rangle \quad \sigma \vdash \langle [b]M, \emptyset\rangle}$$

Fig. 3: Example tableau for the CFPS of Figure 1. M abbreviates $\mu Y.[b]Y$ and N stands for $\nu X.([a,b]X \wedge M)$. Several subtableaux (named $T_{e\,M} \ldots$) are given separately.



$$\dfrac{\dfrac{\dfrac{\dfrac{s_0 \vdash \nu X.\,[a][b]X}{\sigma \vdash \langle \nu X.\,[a][b]X,\ \{\nu X.\,[a][b]X\}\rangle} \qquad \dfrac{}{s_e \vdash \nu X.\,[a][b]X}}{\sigma \vdash \langle [a][b]\nu X.\,[a][b]X,\ \{\nu X.\,[a][b]X\}\rangle} \qquad \dfrac{}{s_e \vdash [a][b]tt}}{\bullet\ \mu \vdash \langle [b]\nu X.\,[a][b]X,\ \{\nu X.\,[a][b]X\}\rangle}}{\epsilon \vdash \langle \nu X.\,[a][b]X,\ \{\nu X.\,[a][b]X\}\rangle} \qquad T \qquad \dfrac{\bullet\ \mu \vdash \langle [b]\nu X.\,[a][b]X,\ \{\nu X.\,[a][b]X\}\rangle}{\ldots}$$

where T is the following (successful) tableau:

$$T = \dfrac{\dfrac{\dfrac{\sigma' \vdash \langle [b]\nu X.\,[a][b]X,\ \{[b]\nu X.\,[a][b]X\}\rangle}{\nu \vdash \langle \nu X.\,[a][b]X,\ \{[b]\nu X.\,[a][b]X\}\rangle}}{\nu \vdash \langle [a][b]\nu X.\,[a][b]X\rangle,\ \{[b]\nu X.\,[a][b]X\}}}{\epsilon' \vdash \langle [b]\nu X.\,[a][b]X,\ \{[b]\nu X.\,[a][b]X\}\rangle}$$

Fig. 4: An example demonstrating the necessity of a more general recurrence rule for ν-formulae

deal with an infinite state space. Suppose we try to prove $\sigma \vdash \langle \mu X.\,\phi,\ \Psi\rangle$, and within the tableau, the same sequent appears again. This means that validity of the sequent implies its validity, a seemingly circular argument. The corresponding tableau in the system from [4, 5] may nevertheless be successful, if an additional condition is met. Let T be the set of copies of σ whose corresponding end state satisfies Ψ. For $s, s' \in T$, let $s' \sqsubset s$ iff the proof steps in the tableau show that $s \in [\![\mu X.\,\phi]\!]$ if $s' \in [\![\mu X.\,\phi]\!]$. Well-foundedness of this order is (roughly) the condition for success. (Intuitively, this is sound, because for the minimal elements w.r.t. this ordering, a successful tableau must contain another argument why they satisfy the formula, which gives the base case of an induction.) But in our tableaux, the order is never well-founded: Either s and s' are in the same copy of the named process graph, then $s = s'$ (introducing a cycle), or s' is in a later copy than s, but then the copying can be repeated infinitely often, again destroying well-foundedness. The order would only be well-founded, if s' were in an earlier copy. But in our tableaux, recurring to an earlier copy is impossible. The proof would have to go through some node with an assertion about the

end state class belonging to $[s]$, and such nodes are leaves in our tableaux. Therefore, recurrence of a sequent with a μ-formula can not be successful in our tableaux.

Indeed, the (safe) recurrence does occur indirectly in our tableaux. Look at the subtableau

$$T_{\rho M} = \dfrac{\dfrac{\rho \vdash \langle \mu Y.[b]Y, \{M, N\}\rangle}{\rho \vdash \langle [b]\mu Y.[b]Y, \{M, N\}\rangle}}{\epsilon \vdash \langle \mu Y.[b]Y, \{M, N\}\rangle}$$

in Fig. 3. The set of $r \in \rho$ with $end(r) \models \{M, N\}$ consists of all copies of ρ ($\{r_0, r_1, \ldots\}$), and the corresponding end states (represented in the last sequent of the tableau) additionally contains s_e.

The validity of $s_e \models \{M, N\}$ is proved at a different place in the global proof, but the small tableau above establishes

$$r_i \models \mu Y.[b]Y \quad \text{if} \quad r_{i-1} \models \mu Y.[b]Y$$

This is exactly what would require a recurrence of $\mu Y.[b]Y$ in the system of [4, 5].

In the remainder of the paper we are going to show that the construction of successful tableaux is an effective, sound and complete verification method for context–free processes. Due to the presence of recurrences both in processes and formulae, the proofs are rather complicated, and require some preparations.

Let us start with two basic observations concerning our tableau rules. On the one hand, each rule is sound in that the sequents below the line always imply the one above. On the other hand, there is always a rule which can be applied if a sequent is valid and the information about the end state is complete, i.e. if the sequent is exact.

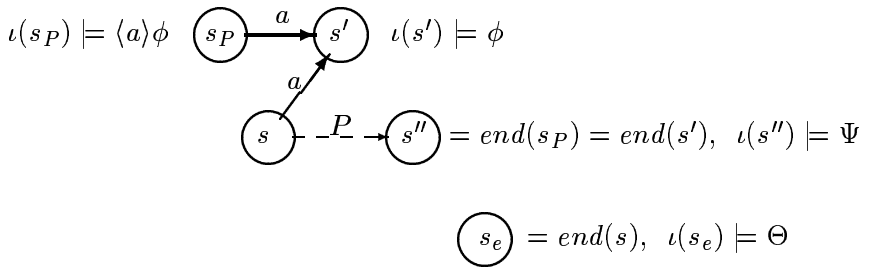
LEMMA 1. *Each rule is sound, i.e. if the sequents below the line are valid, then so is the one above the line.*

PROOF. This is rather obvious for all rules except the Modality Rules. Here, we present the proof for the Diamond Rules.

For the proof of the first Diamond Rule let $\sigma' \vdash \langle \phi, \Theta\rangle$ be valid and let $\sigma \xrightarrow{a} \sigma'$ be a transition in a PPG named by P. Then we have to show that $\sigma \vdash \langle\langle a\rangle\phi, \Theta\rangle$ is valid. Let $s \in FExp_{\mathcal{P}}(P)$ be the first copy of σ, i.e. $s = \mathit{first\_copy}(\sigma)$, and let s' be either the first copy, $\mathit{first\_copy}(\sigma')$, of σ', or s_e in case of $\sigma' = \epsilon_P$. Taking a faithful embedding ι of $FExp_{\mathcal{P}}(P)$ into some PG G with $\iota(s_e) \models_G \Theta$, it remains to show $\iota(s) \models_G \langle a\rangle\phi$.

The validity of $\sigma' \vdash \langle \phi, \Theta\rangle$ implies $\iota(s') \models_G \phi$, and therefore, since $\sigma \xrightarrow{a} \sigma'$, also $\iota(s) \xrightarrow{a} \iota(s')$ and $\iota(s) \models_G \langle a\rangle\phi$ as desired.

The second Diamond Rule covers the case where the a-step is executed within a procedure. Let $\sigma_P \vdash \langle\langle a\rangle\phi, \Psi\rangle$ and $\sigma'' \vdash \langle\psi, \Theta\rangle$ for all $\psi \in \Psi$ be valid, let $\sigma \xrightarrow{P} \sigma''$ be a named transition in a PPG Q. Moreover let s and s'' (or s_e, if $\sigma'' = \epsilon_Q$) be the first copies of σ resp. σ'' in $FExp_{\mathcal{P}}(Q)$, and let s_P be the copy of σ_P created when faithfully expanding $\sigma \xrightarrow{P} \sigma''$. This is the situation we are in (in fact, $s = s''$ or $s'' = s_e$ may hold, not affecting much our argumentation).



Now taking a faithful embedding ι of $FExp_{\mathcal{P}}(Q)$ into some PG G with $\iota(s_e) \models_G \Theta$, we have to show $\iota(s) \models_G \langle a\rangle\phi$. ι also induces a faithful embedding of $FExp_{\mathcal{P}}(P)$ into G with start state $\iota(s_P)$ and end state $\iota(s'')$. $\iota(s'')$ satisfies Ψ, because all $\sigma'' \vdash \langle\psi, \Theta\rangle$ are valid, and the assumption $\sigma_P \vdash \langle\langle a\rangle\phi, \Psi\rangle$ guarantees that $\iota(s_P)$ satisfies $\langle a\rangle\phi$, i.e. $\iota(s_P)$ has an a-derivative s' satisfying ϕ. Since any derivative of $\iota(s_P)$ is also a derivative of $\iota(s)$ (by the definition of the faithful expansion), we obtain $\iota(s) \models_G \langle a\rangle\phi$, which completes the proof of the second Diamond Rule.

Proving the soundness of the Box Rules is quite similar. □

The existence of exact sequents is the key to the completeness of our method. The following lemma states that it is possible to expand the proof tree while preserving exactness.

LEMMA 2. *For every exact sequent there is a tableau with the sequent at its root which is either*

- *successful, or*
- *contains at least one nontrivial rule application and its unsuccessful leaves are exact sequents.*

PROOF. Let the sequent be $\sigma \vdash \langle\psi, \Theta\rangle$ and let $s \in \sigma$ with $s \models \psi$ and $\Theta = \{\, \theta \in CL(\psi) \mid end(s) \models \theta \,\}$. The result follows from a straightforward case analysis.

$\psi = \psi_1 \vee \psi_2$**:** Then $s \models \psi_i$ for some $i \in \{1, 2\}$. Let $\Theta' =_{\text{df}} \Theta \cap CL(\psi_i)$. Then $\sigma \vdash \langle\psi_i, \Theta'\rangle$ is exact, and an application of the Disjunction Rule and maybe the Weakening Rule provide the required tableau.

$\psi = \langle a\rangle\psi'$**:** . Then $s \models \psi$ implies that either $\sigma \xrightarrow{a} \sigma'$ or $\sigma \xrightarrow{P} \sigma''$ for a PPG P with $\sigma_P \xrightarrow{a} \sigma'$. In the first case, also $\sigma' \vdash \langle\psi', \Theta \cap CL(\psi')\rangle$ is exact,

and we can apply the first Diamond Rule (and maybe Weakening). In the second case, let s' be the relevant copy of σ_P, and let $\Theta' = \{\, \theta \in CL(\psi) \mid end(s') \models \theta \,\}$. Then $\sigma_P \vdash \langle \psi, \Theta' \rangle$ is exact, and we can similarly construct exact sequents for the relevant copy of σ'' and all elements of Θ'. Finally, $\sigma \vdash \langle \psi, \Theta \rangle$ can be reduced to these sequents by applying the second Diamond Rule.

The case of a box formula is similar, and all others are obvious. □

The general structure of the proof of soundness and completeness of the tableau system is an induction on the size of the formula. More precisely, it proceeds by induction on the size of the *closure* of the formula and establishes soundness and completeness in parallel.

One of the problems we face in formulating the proof is that of relating the derivability of intermediate assertions, which are second-order, to the semantics of mu-calculus formulae, which gives a set of (first-order) states. In the tableaux, we compute second-order fixpoints, but the semantics of formulae is given by first-order fixpoints. To be able to formulate properties suitable for inductive proofs, we introduce the notion of *first-order derivability*. This notion allows to speak of the set of formulae which are derivable for some state of the expansion of a CFPS. Note that from Definition 9 we only get derivability either of formulae for the start state or of second-order assertions for state classes.

DEFINITION 10. *Let s be a state different from the global end state. A closed formula ϕ is* first-order derivable *for s if there is a derivable sequent $[s] \vdash \langle \phi, \Theta \rangle$ where $\Theta \subseteq CL(\phi)$ and all $\theta \in \Theta$ are first-order derivable for $end(s)$. ϕ is* first-order derivable *for the global end state s_e, if $s_e \vdash \phi$ is derivable (via the End Rules). The set of states for which ϕ is first-order derivable is denoted by $[\![\phi]\!]$.*

This recursive definition is well-founded: $end(s)$ does not belong to the same copy as s, it is created at least one expansion step earlier.

Another difficulty in the proof arises because the inductive definition of the semantics of formulae involves formulae with free variables and an environment. In the tableaux, no free variables do occur. Variables are always replaced by the closed μ- or ν-formula to which they are bound. Thus we have to relate the subformulae and the environment in the semantic definition to the substitution instances in the tableaux. Note that the substitutions to be done for any subformula of a given closed formula are determined by its context.

DEFINITION 11. *Let $\phi \in \mathcal{F}$ and let ψ be a subformula occurrence in ϕ. Then $\widetilde{\psi}$ denotes the closed formula resulting from ϕ by iteratively substituting the appropriate fixpoint formula for every free variable occurrence.*

As an example, consider the formula $\nu X.\langle a \rangle(\nu Y.X \vee [b]Y)$. Then

$$\widetilde{[b]Y} \;=\; [b]\,\nu Y.(\widetilde{X} \vee [b]Y) \;=\; [b]\,\nu Y.((\nu X.\langle a \rangle(\nu Y.X \vee [b]Y)) \vee [b]Y) \;.$$

Up to renaming of bound variables, $\widetilde{\psi}$ is unique. In the following, we will assume a distinguished variable naming, i.e. that different fixpoint definitions use different variable names. This leads to a globally well-defined function $\widetilde{\ }$. In particular, $\widetilde{X}$ does not depend on the position of the occurrence of X in ϕ.

The following two lemmata give a handle to proving the completeness for minimal fixpoint formulae and the soundness for maximal fixpoint formulae.

LEMMA 3. *Let ψ be a subformula of $\phi \in \mathcal{F}$ where all free variables are bound to minimal fixpoints in ϕ. Then we have $[\![\widetilde{\psi}]\!] \subseteq [\![\widetilde{\psi}]\!]$, whenever $[\![\nu X.\,\Omega]\!] \subseteq [\![\nu X.\,\Omega]\!]$ for any closed subformula $\nu X.\,\Omega$ of ψ.*

PROOF. Let $X_1, \ldots, X_n$ contain all (free and bound) variables in ψ which are not in the scope of a maximal fixpoint operator, and $\widetilde{e}$ denote $\{X_1 \mapsto [\![\widetilde{X_1}]\!], \ldots, X_n \mapsto [\![\widetilde{X_n}]\!]\}$. Then it suffices to prove

$$[\![\psi]\!]\{X_1 \mapsto [\![\widetilde{X_1}]\!], \ldots, X_n \mapsto [\![\widetilde{X_n}]\!]\} \subseteq [\![\widetilde{\psi}]\!] \quad ,$$

by induction on the structure of ψ.

The central argument is presented in the case where $\psi = \mu X.\,\psi'$ below. The rest is a straightforward, but, in particular in the case of modalities, tedious case distinction.

$\psi = X_i$**:** Then $[\![\psi]\!]\widetilde{e} = [\![\widetilde{X_i}]\!] = [\![\widetilde{\psi}]\!]$.

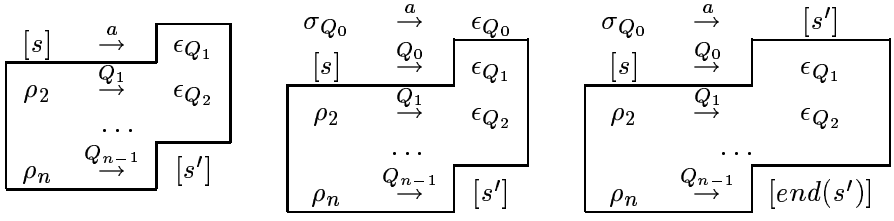
$\psi = \psi_1 \vee \psi_2$**:** In this case $[\![\psi]\!]\widetilde{e} = [\![\psi_1]\!]\widetilde{e} \cup [\![\psi_2]\!]\widetilde{e} \subseteq [\![\widetilde{\psi_1}]\!] \cup [\![\widetilde{\psi_2}]\!]$ by induction. Thus, for $s \in [\![\psi]\!]\widetilde{e}$ there is a $\Theta \subseteq CL(\widetilde{\psi_1})$ or $\Theta \subseteq CL(\widetilde{\psi_2})$ with $end(s) \in [\![\Theta]\!]$ and a successful tableau for $[s] \vdash \langle\widetilde{\psi_1}, \Theta\rangle$ resp. $[s] \vdash \langle\widetilde{\psi_2}, \Theta\rangle$.

An application of the Disjunction Rule gives a tableau for $[s] \vdash \langle\widetilde{\psi}, \Theta\rangle$, and of course $\Theta \subseteq CL(\widetilde{\psi})$.

$\psi = \psi_1 \wedge \psi_2$**:** By induction, we get two tableaux with 'independent' sets of formulae Θ_1 and Θ_2. Applying the Weakening Rule with $\Theta =_{\text{df}} \Theta_1 \cup \Theta_2$ followed by the Conjunction Rule yields the desired result.

$\psi = \langle a\rangle\psi'$**:** Let $s \in [\![\psi]\!]\widetilde{e}$ and $s' \in [\![\psi']\!]\widetilde{e}$ with $s \xrightarrow{a} s'$. Then we know by induction that $s' \in [\![\widetilde{\psi'}]\!]$. Thus there exists a $\Theta \subseteq CL(\widetilde{\psi'})$ with $end(s') \in [\![\theta]\!]$ for $\theta \in \Theta$ and a derivation of $[s'] \vdash \langle\widetilde{\psi'}, \Theta\rangle$.

In the CFPS, three cases need to be distinguished:



These cases are now treated successively. Each of them splits into two subcases, requiring different argumentations: Either $n = 1$ and the part within the frame is missing, or $n > 1$, and the frame is nontrivial. For the simple subcase of the first, where $[s] \xrightarrow{a} [s']$, we have $end(s) = end(s')$, and we can extend the derivation of $[s'] \vdash \langle \widetilde{\psi'}, \Theta \rangle$ to a derivation of $[s] \vdash \langle \widetilde{\langle a \rangle \psi'}, \Theta \rangle$ by one application of the first Diamond Rule. Thus let $n > 1$. Then $s' = end(s)$, and with the first Diamond Rule we can derive $[s] \vdash \langle \widetilde{\langle a \rangle \psi'}, \{\widetilde{\psi'}\} \rangle$ from the successful leaf $\epsilon_{Q_1} \vdash \langle \widetilde{\psi'}, \{\widetilde{\psi'}\} \rangle$, which completes the proof of the first case.

In the second case, we can derive $\sigma_{Q_0} \vdash \langle \widetilde{\langle a \rangle \psi'}, \{\widetilde{\psi'}\} \rangle$. The rest is analogous to the first case, with the only difference that the second Diamond Rule is applied instead of the first one, which requires the additional sequent established above.

In the third case, we can derive $\sigma_{Q_0} \vdash \langle \widetilde{\langle a \rangle \psi'}, \Theta \rangle$.
In the subcase for $n = 1$ we have $end(s) = end(end(s'))$, and since $end(s') \in [\![\theta]\!]$ for $\theta \in \Theta$, there is (by the definition of $[\![\cdot]\!]$) a derivation of $[end(s')] \vdash \langle \theta, \Theta' \rangle$ for some $\Theta' \subseteq CL(\widetilde{\langle a \rangle \psi'})$ with $end(end(s')) \in [\![\Theta']\!]$ ($\Rightarrow$ $end(s) \in [\![\Theta']\!]$). This and the assertion about σ_{Q_0} allow us to derive $[s] \vdash \langle \widetilde{\langle a \rangle \psi'}, \Theta' \rangle$ as desired.
Thus let $n > 1$ and therefore $end(s) = end(s')$. The sequents $\epsilon_{Q_1} \vdash \langle \theta, \Theta \rangle$ for $\theta \in \Theta$ are successful leaves, and combining them with $\sigma_{Q_0} \vdash \langle \widetilde{\langle a \rangle \psi}, \Theta \rangle$ gives a derivation of $[s] \vdash \langle \langle a \rangle \psi, \Theta \rangle$ by means of the second Diamond Rule. Since $end(s) = end(s')$, also $end(s) \in [\![\Theta]\!]$ by our assumptions, completing the argument.

$\psi = [a]\psi'$**:** All derivatives of $[s]$ with $s \in [\![\psi]\!]\widetilde{e}$ have to be considered, leading to similar subcases as above, plus cases where there is no a-derivative at all.

$\psi = \nu X.\psi'$**:** By assumption, only variables bound to minimal fixpoints are free in ψ. Thus ψ must be closed, because ϕ is alternation-free. Thus we are done simply by means of the assumption $[\![\nu X.\psi']\!] \subseteq [\![\nu X.\psi']\!]$.

$\psi = \mu X.\psi'$**:** The induction hypothesis gives us

$$[\![\widetilde{\psi'}]\!] \supseteq [\![\psi']\!]\{X_1 \mapsto [\![\widetilde{X_1}]\!], \ldots, X_n \mapsto [\![\widetilde{X_n}]\!], X \mapsto [\![\widetilde{X}]\!]\}$$

and since $[\![\widetilde{X}]\!] = [\![\widetilde{\psi'}]\!]$, we also have $[\![\widetilde{\psi'}]\!] \supseteq [\![\psi']\!]\widetilde{e}[X \mapsto [\![\widetilde{\psi'}]\!]]$. Thus, $[\![\widetilde{\psi'}]\!]$ is decreased by $\lambda S'.[\![\psi']\!]\widetilde{e}[X \mapsto S']$. Now considering the definition of $[\![\mu X.\psi']\!]$ yields the required inclusion $[\![\widetilde{\psi'}]\!] \supseteq [\![\mu X.\psi']\!]\widetilde{e}$. □

LEMMA 4. *Let ψ be a subformula of $\phi \in \mathcal{F}$ where all free variables are bound to maximal fixpoints in ϕ. Then we have $[\![\widetilde{\psi}]\!] \supseteq [\![\widetilde{\psi}]\!]$, whenever $[\![\mu X.\Omega]\!] \supseteq [\![\mu X.\Omega]\!]$ for any closed subformula $\mu X.\Omega$ of ψ.*

PROOF. Let $X_1, \ldots, X_n$ contain all (free and bound) variables in ψ which are not in the scope of a minimal fixpoint operator, and $\widetilde{e}$ denote $\{X_1 \mapsto [\![\widetilde{X_1}]\!], \ldots, X_n \mapsto [\![\widetilde{X_n}]\!]\}$. Then, as in Lemma 3, we prove

$$[\![\psi]\!]\{X_1 \mapsto [\![\widetilde{X_1}]\!], \ldots, X_n \mapsto [\![\widetilde{X_n}]\!]\} \supseteq [\![\widetilde{\psi}]\!] \quad ,$$

by induction on the structure of ψ. In this case the central argument is needed for $\psi = \nu X.\, \psi'$.

$\psi = X_i$: Then $[\![\psi]\!]\widetilde{e} = [\![\widetilde{X_i}]\!] = [\![\widetilde{\psi}]\!]$.

$\psi = \psi_1 \vee \psi_2$: In this case $[\![\psi]\!]\widetilde{e} = [\![\psi_1]\!]\widetilde{e} \cup [\![\psi_2]\!]\widetilde{e} \supseteq [\![\widetilde{\psi_1}]\!] \cup [\![\widetilde{\psi_2}]\!]$ by induction. Let $s \in [\![\widetilde{\psi_1} \vee \widetilde{\psi_2}]\!]$. Then there is a derivation of $[s] \vdash \langle \widetilde{\psi_1} \vee \widetilde{\psi_2}, \Theta \rangle$ for some $end(s) \in [\![\Theta]\!]$. The Or Rule must be applied in this derivation to decompose $\widetilde{\psi_1} \vee \widetilde{\psi_2}$, possibly after doing weakening. So there is a subtableau for $[s] \vdash \langle \psi_i, \Theta' \rangle$ for either $i = 1$ or $i = 2$ with $\Theta' \subseteq \Theta$. If there is no leaf recurring to $[s] \vdash \langle \widetilde{\psi_1} \vee \widetilde{\psi_2}, \Theta \rangle$, the subtableau is already a complete derivation. Otherwise, we get a derivation by extending the subtableau at the recurring leaves. This proves $s \in [\![\widetilde{\psi_i}]\!]$. From $[\![\widetilde{\psi_i}]\!] \subseteq [\![\psi]\!]\widetilde{e}$ above we conclude $s \in [\![\psi]\!]\widetilde{e}$, as required.

$\psi = \psi_1 \wedge \psi_2$: similar.

$\psi = \langle a \rangle \psi'$: Let $s \in [\![\langle a \rangle \widetilde{\psi'}]\!]$. Then $[s] \vdash \langle \langle a \rangle \widetilde{\psi'}, \Theta \rangle$, where $end(s) \in [\![\Theta]\!]$, is derived by an application of one of the diamond rules. By reasoning similar to that in the proof of Lemma 1 (compare also the cases in Lemma 3), we get some s' with $s \xrightarrow{a} s'$ for an s' with the following possibilities:

- $end(s) = end(s')$ and $[s'] \vdash \langle \widetilde{\psi'}, \Theta \rangle$ is derivable.
- $end(s) = s'$ and $\psi' \in \Theta$. Again, $s' \in [\![\widetilde{\psi'}]\!]$.
- $end(s) = end(end(s'))$, and $[s'] \vdash \langle \widetilde{\psi'}, \Theta' \rangle$ and $[end(s')] \vdash \langle \Theta', \Theta \rangle$ are derivable for some Θ' .
- $end(s) = end(s')$, and $[s'] \vdash \langle \widetilde{\psi'}, \Theta' \rangle$ is derivable for some $\Theta' \subseteq \Theta$.

The first two cases may arise if the first or second rule is applied, the last two can only occur from the second. In all cases, $s' \in [\![\widetilde{\psi'}]\!]$ can be deduced. From the induction hypothesis and the semantics definition $s \in [\![\psi]\!]\widetilde{e}$ follows as required.

$\psi = [a]\psi'$: similar.

$\psi = \mu X.\psi'$: by the assumptions.

$\psi = \nu X_i.\psi'$: Let $s \in [\![\widetilde{\psi}]\!] = [\![\widetilde{X_i}]\!]$. Then the maximal fixpoint rule must have been applied and there is a derivation for $[s] \vdash \langle \widetilde{\psi'}, \Theta \rangle$ where $end(s) \in [\![\Theta]\!]$. By induction, we get $s \in [\![\psi']\!]\widetilde{e}$. Thus,

$$[\![\widetilde{X_i}]\!] \subseteq [\![\psi']\!]\{X_1 \mapsto [\![\widetilde{X_1}]\!], \ldots, X_i \mapsto [\![\widetilde{X_i}]\!], \ldots, X_n \mapsto [\![\widetilde{X_n}]\!]\} \ ,$$

which implies

$$[\![\widetilde{\psi}]\!] = [\![\widetilde{X_i}]\!] \subseteq [\![\nu X_i.\psi']\!]\widetilde{e}$$

by the definition of the semantics of maximal fixpoints.□

Now we are in a position to prove the main result of this paper, the soundness and completeness of the tableau system.

THEOREM 1. (SOUNDNESS AND COMPLETENESS) *A formula is derivable for a CFPS iff it is valid.*

PROOF. Soundness and completeness of the end rules are easy to show. So we concentrate on second-order sequents. We prove that every derivable sequent $\sigma \vdash \langle\widetilde{\psi}, \Theta\rangle$ is valid (soundness) and that every *exact* sequent $\sigma \vdash \langle\widetilde{\psi}, \Theta\rangle$ is derivable (completeness). The proof proceeds by induction on $|CL(\widetilde{\psi})|$.

$\widetilde{\psi}$ may contain fixpoint formulae θ with $\widetilde{\psi} \in CL(\widetilde{\theta})$. Since we are dealing with the alternation-free mu-calculus, these are either all maximal or all minimal fixpoint formulae. Accordingly, we call ψ a minimal or a maximal formula. These are treated separately in the proof. Both cases subsume formulae which are not in the closure of any subformula, i.e. which can be viewed as being both a minimal and a maximal formula.

ψ is maximal: Let $\sigma \vdash \langle\widetilde{\psi}, \Theta\rangle$ be an exact sequent. By Lemma 2, there is a nontrivial tableau reducing this sequent to other exact sequents. For those sequents concerning formulae with a smaller closure, there is by induction a successful tableau which we can add. Otherwise, the formula at the leaf has the same closure as $\widetilde{\psi}$, and we keep on extending the leaves. Since the closure is not increased, only finitely many sequents can occur. So on every path a sequent must eventually recur. The recurring leaf is successful, since only the unfolding of a maximal fixpoint can be responsible for its recurrence. Thus, $\sigma \vdash \langle\widetilde{\psi}, \Theta\rangle$ is derivable.

Let $\sigma \vdash \langle\widetilde{\psi}, \Theta\rangle$ be derivable, and let $s \in \sigma$ with $end(s) \in [\![\Theta]\!]$. Using the induction hypothesis for formulae $\theta \in \mathcal{F}$ with $CL(\theta) \subset CL(\widetilde{\psi})$ and the already established completeness for formulae $\theta \in \mathcal{F}$ with $CL(\theta) = CL(\widetilde{\psi})$ (which cannot be 'minimal') yields $[\![\theta]\!] \subseteq [\![\theta]\!]$ for such θ . Therefore, $end(s) \in [\![\Theta]\!]$ and $s \in [\![\widetilde{\psi}]\!]$. Lemma 4 allows us to conclude $s \in [\![\widetilde{\psi}]\!]$. Thus, the sequent $\sigma \vdash \langle\widetilde{\psi}, \Theta\rangle$ is valid.

ψ is minimal: The validity of every derivable sequent follows from the induction hypothesis and Lemma 1, because every successful tableau must reduce a minimal formula to formulae with smaller closure.

Let $\sigma \vdash \langle\widetilde{\psi}, \Theta\rangle$ be exact. Let $s \in [\![\widetilde{\psi}]\!]$ with $\Theta = \{\, \theta \in CL(\widetilde{\psi}) \mid end(s) \in [\![\theta]\!] \,\}$ be given. Lemma 3 yields $s \in [\![\widetilde{\psi}]\!]$, i.e. there is a derivation of $\sigma \vdash \langle\widetilde{\psi}, \Theta'\rangle$ where $end(s) \in [\![\Theta']\!]$. Using the soundness established so far, an induction on the expansion depth shows $[\![\widetilde{\theta}]\!] \subseteq [\![\widetilde{\theta}]\!]$ for any $\theta \in \mathcal{F}$ with

$CL(\theta) \subseteq CL(\widetilde{\psi})$. Thus, $end(s) \in [\![\Theta']\!]$, which implies $\Theta' \subseteq \Theta$. So we can apply the Weakening Rule to derive $\sigma \vdash \langle \widetilde{\psi}, \Theta \rangle$.

This establishes soundness and completeness for second-order sequents. Since the end rules share these properties, the theorem follows. □

A CFPS has only finitely many state classes and the closure of a formula ϕ is a finite set. Thus only a finite number of sequents and a finite number of intermediate formulae need to be considered during the construction of a tableau. This yields:

COROLLARY 1. (EFFECTIVENESS) *The tableau system provides an effective sound and complete model checking procedure for context–free processes and alternation–free formulae.*

4. Conclusions and Future Work

We have presented a local model checking algorithm that decides the alternation-free modal mu-calculus for *context-free* processes. Heart of this algorithm is a purely syntactic sound and complete formal system, which in contrast to the known tableau techniques, uses intermediate second-order assertions. These assertions provide a finite representation of all the infinite state sets which may arise during the proof in terms of the finite representation of the context-free argument process. This finiteness is the key to the effectiveness of our local model checking procedure.

Yet more can be done than deciding alternation-free formulae for context-free processes. Muller and Schupp [20] proved that Monadic Second Order Logic (MSOL) is decidable for pushdown transition graphs. Pushdown transition graphs are a strict generalization of context-free processes with respect to bisimulation semantics, and even the full mu-calculus can be embedded into MSOL. Also, Burkart and Steffen [7] considered pushdown transition graphs, and Purushothaman [21] and Hungar [15] further generalized the process language by allowing processes to be called with process parameters.

The most challenging question we are currently considering is the extension of our techniques covering the full modal mu-calculus. The decision procedure of Muller and Schupp for MSOL is non-elementary and thus not applicable to practical problems. Bradfield, Esparza and Stirling are working on an adaptation of the tableaux method of [4, 5] based on success conditions requiring a global investigation of the proof tree. This method is extremely complex, and it is not yet clear how much this approach improves on the complexity of Muller and Schupp's algorithm [3]. We conjecture that other techniques would lead to an elementary algorithm via a locally decidable condition for success.

Finally, we are looking at a modification of our method to improve the worst-case time complexity along the lines of [1, 19]: by keeping track of intermediate results it should be straightforward to almost arrive at the

worst case time complexity of the iterative algorithm of [6] instead of the double exponential complexity of a straightforward algorithm for tableaux construction.

Acknowledgements: We thank Olaf Burkart, Gerald Lüttgen, and Colin Stirling for discussions and comments on previous versions of this paper.

References

[1] Andersen, H., *Model checking on boolean graphs.* ESOP '92, LNCS 582 (1992), 1–19.
[2] Baeten, J.C.M., Bergstra, J.A., and Klop, J.W.. *Decidability of Bisimulation Equivalence for Processes Generating Context-Free Languages.* Technical Report CS-R8632, CWI, September 1987.
[3] Bradfield, J.C., Esparza, J., and Stirling, C.P., private communication.
[4] Bradfield, J.C., *Verifying temporal properties of systems.* Birkhäuser, Boston (1992).
[5] Bradfield, J.C., and Stirling, C.P., *Verifying temporal properties of processes.* Proc. CONCUR '90, LNCS 458 (1990), 115-125.
[6] Burkart, O., and Steffen, B., *Model checking for context-free processes.* CONCUR '92, LNCS 630 (1992), 123–137.
[7] Burkart, O., and Steffen, B., *Pushdown processes: Parallel composition and model checking.* CONCUR '94, LNCS 836 (1992), 98–113.
[8] Clarke, E.M., Emerson, E.A., and Sistla, A.P., *Automatic verification of finite state concurrent systems using temporal logic specifications.* ACM TOPLAS **8** (1986), 244–263.
[9] Cleaveland, R., *Tableau-based model checking in the propositional mu-calculus.* Acta Inf. **27** (1990), 725-747.
[10] Caucal, D. and Monfort, R. *On the Transition Graphs of Automata and Grammars.* In *Graph-Theoretic Concepts in Computer Science*, LNCS 484 (1990), 311–337.
[11] Cleaveland, R., Parrow, J., and Steffen, B., *The concurrency workbench.* Workshop Automatic Verification Methods for Finite-State Systems, LNCS 407 (1989), 24–37.
[12] Cleaveland, R., and Steffen, B., *Computing behavioral relations, logically.* ICALP '91, LNCS 510 (1991), 127–138.
[13] Cleaveland, R., and Steffen, B., *A linear-time model-checking algorithm for the alternation-free modal mu-calculus.* CAV 91, LNCS 575 (1992), 48–58.
[14] Emerson, E.A., and Lei, C.-L., *Efficient model checking in fragments of the propositional mu-calculus.* 1st LiCS (1986), 267–278.
[15] Hungar, H. *Model checking of macro processes,* CAV '94, LNCS 818 (1994), 169–181.
[16] Huynh, D.T., and Tian, L., *Deciding bisimilarity of normed context-free processes is in* Σ_2^p. Tech. Rep. UTDCS-1-92, Univ. Texas Dallas (1992).
[17] Kozen, D., *Results on the propositional* μ*-calculus.* TCS **27** (1983), 333–354.
[18] Larsen, K.G., *Proof systems for satisfiability in Hennessy-Milner logic with recursion.* TCS **72** (1990), 265-288.
[19] Larsen, K.G., *Efficient local correctness checking.* CAV '92, LNCS 663, 30–44.
[20] Muller, D.E. and Schupp, P.E. *The Theory of Ends, Pushdown Automata, and Second-Order Logic.* TCS **37** (1985), 51–75.
[21] Purushothaman Iyer, S., *A note on model checking context-free processes.* NAPAW '93, to appear in LNCS.
[22] Stirling, C.P., and Walker, D.J., *Local model checking in the modal mu-calculus.* TAPSOFT '89, LNCS 351 (1989), 369-383.
[23] Winskel, G., *A note on model checking the modal mu-calculus.* ICALP '89, LNCS 372 (1989), 761–772.