BRICS
Basic Research in Computer Science

BRICS RS-98-40 Andersen et al.: A Compositional Proof System for the Modal μ-Calculus

A Compositional Proof System for the Modal μ-Calculus

Henrik Reif Andersen
Colin Stirling
Glynn Winskel

BRICS Report Series RS-98-40

ISSN 0909-0878 December 1998

See back inner page for a list of recent BRICS Report Series publications.
Copies may be obtained by contacting:

BRICS
Department of Computer Science
University of Aarhus
Ny Munkegade, building 540
DK–8000 Aarhus C
Denmark
Telephone: +45 8942 3360
Telefax: +45 8942 3255
Internet: BRICS@brics.dk

BRICS publications are in general accessible through the World Wide Web and anonymous FTP through these URLs:

`http://www.brics.dk`
`ftp://ftp.brics.dk`
This document in subdirectory `RS/98/40/`

A Compositional Proof System for the Modal μ-Calculus*

Henrik Reif Andersen†
Department of Computer Science
Technical University of Denmark
DK-2800 Lyngby
Denmark

hra@id.dtu.dk

Colin Stirling
Laboratory for Foundations of Computer Science
University of Edinburgh
Edinburgh EH9 3JZ
Scotland, UK
cps@dcs.edinburgh.ac.uk

Glynn Winskel
BRICS‡
Department of Computer Science
Aarhus University
DK-8000 Aarhus C
Denmark
gwinskel@brics.dk

Abstract

We present a proof system for determining satisfaction between processes in a fairly general process algebra and assertions of the modal μ-calculus. The proof system is compositional in the structure of processes. It extends earlier work on compositional reasoning within the modal μ-calculus and combines it with techniques from work on local model checking. The proof system is sound for all processes and complete for a class of finite-state processes.

1 Introduction

The propositional μ-calculus of Kozen [Kozen, 1983] which was introduced as a powerful extension of propositional dynamic logic has received growing interest as a logic for concurrent systems. This is mainly due to the expressiveness of the logic, which is known to subsume many modal and temporal logics, and the fact that very few operators are needed in achieving this: The logic is an extension of relativized, minimal modal logic K – also known as

*Extended abstract appears in: Proceedings of LICS'94, IEEE Computer Society Press.
†Supported by the Danish Technical Research Council.
‡Basic Research in Computer Science, Centre of the Danish National Research Foundation.

Hennessy-Milner logic in the process algebra community – with minimum and maximum fixed points. It is due to this connection (explained in more detail in [Stirling, 1992]) that we use the name the *modal* μ-calculus.

It is customary to consider Kripke models or, equivalently, labelled transition systems as models for interpretation of the logic. Since labelled transition systems are used in giving operational semantics of process languages, it is straightforward to view the modal μ-calculus as a language for expressing properties of processes. Despite the expressiveness, it turns out that validity is decidable for the modal μ-calculus, and for finite-state processes the problem of deciding *satisfaction* between a process and an assertion is decidable too. A range of algorithms and proof systems for this problem has been given in the literature, e.g. [Emerson and Lei, 1986, Arnold and Crubille, 1988, Larsen, 1988, Stirling and Walker, 1991, Cleaveland, 1990, Winskel, 1989, Cleaveland and Steffen, 1992, Andersen, 1994, Vergauwen and Lewi, 1992, Larsen, 1992, Cleaveland et al., 1992, Andersen, 1993]. They mostly rely on globally or locally computing the underlying transition system. However, what we seek here is a method that is *compositional* in the structure of processes, and which does not rely on computing the underlying transition system.

Compositionality is important for at least the following reasons. Firstly, it makes the verification *modular*, so that when changing part of a system only the verification concerning that particular part must be redone. Secondly, when designing a system or *synthesising* a process the compositionality makes it possible to have undefined parts of a process and still be able to reason about it. For instance, it might be possible to reveal inconsistencies in the specification or prove that with the choices already taken in the design no component supplied for the missing parts will ever be able to make the overall system satisfy the original specification. Thirdly, it makes it possible to *decompose* the verification task into potentially simpler tasks. Finally, it can make possible the *reuse* of verified components; their previous verification can be used to show that they meet the requirements on the components of a larger system.

Our method will be a *compositional proof system*, sound for arbitrary processes and complete for a class of finite-state processes. The proof system is compositional in the sense of the rules being guided by the structure of processes and not looking into the structure of subprocesses. Earlier work on compositional proof systems related to the modal μ-calculus includes work by Stirling [Stirling, 1985b, Stirling, 1985a, Stirling, 1987], Winskel [Winskel, 1985, Winskel, 1986, Winskel, 1990a, Winskel, 1990b], Larsen and Xinxin [Larsen and Xinxin, 1990], Andersen and Winskel [Andersen and Winskel, 1992]. The proof system presented here is along the lines of the work by Stirling and Winskel, but it extends their early work for Hennessy-Milner logic to a proper treatment of recursive processes and the full modal μ-calculus. It also gives new rules for parallel composition and the other static operators. Actually, to a certain extent, the system can be seen as a result of turning the operational reductions of Larsen and Xinxin and the syntactic reductions of Andersen and Winskel into proof rules. But the match is not exact; apart from the new static rules the treatment of fixed points is closer to the work on local model checking [Larsen, 1988, Stirling and Walker, 1991, Cleaveland, 1990, Winskel, 1989].

2 Languages

$$p \stackrel{*}{\to} p \qquad a.p \stackrel{a}{\to} p$$

$$\frac{p \stackrel{\alpha}{\to} p'}{p+q \stackrel{\alpha}{\to} p'}\ \alpha \neq * \qquad \frac{q \stackrel{\alpha}{\to} q'}{p+q \stackrel{\alpha}{\to} q'}\ \alpha \neq * \qquad \frac{t[rec\ x.t/x] \stackrel{\alpha}{\to} t'}{rec\ x.t \stackrel{\alpha}{\to} t'}\ \alpha \neq *$$

$$\frac{p \stackrel{\alpha}{\to} p' \quad q \stackrel{\beta}{\to} q'}{p \times q \stackrel{\alpha\times\beta}{\to} p' \times q'} \qquad \frac{p \stackrel{\alpha}{\to} p'}{p\{\Xi\} \stackrel{\beta}{\to} p'\{\Xi\}}\ \Xi(\alpha) = \beta \qquad \frac{p \stackrel{\alpha}{\to} p'}{p \upharpoonright \Lambda \stackrel{\alpha}{\to} p' \upharpoonright \Lambda}\ \alpha \in \Lambda$$

Table 1: Operational rules.

The process language has a general parallel composition operator called a *product*, $t_0 \times t_1$, that allows the components to proceed both synchronously and asynchronously. Synchronization can then be enforced – or disallowed – through a *restriction operator* and synchronized actions can be given proper names through a *relabelling operator*. We refrain from giving details of how this allows a wide range of parallel operators to be encoded (see for example [Winskel, 1984] or [Andersen, 1993]), and we stick to introducing the language.

Let *Act* be a set of *basic actions* not containing the *idling action* $*$. The set of *composite actions* Act_* is the free $*, \times$-algebra over $Act \cup \{*\}$ such that $* \times * = *$. We let $a, b, \ldots$ range over basic actions, $\alpha, \beta, \ldots$ over composite actions, and κ over sets of composite actions. The set of process terms are generated from the grammar:

$$t ::= \mathbf{0} \mid a.t \mid t_0 + t_1 \mid t_0 \times t_1 \mid t\{\Xi\} \mid t \restriction \Lambda \mid x \mid rec\ x.t$$

The term constructors are called: *nil*, *prefix*, *sum*, *product*, *relabelling*, *restriction*, *process variable*, and *recursion*. The *restricting set* Λ is any subset of Act_* containing $\{*\}$; the *relabelling function* $\Xi : Act_* \to Act_*$ must be strict and injective on idling actions, i.e. $\Xi(*) = *$ and $\Xi^{-1}(*) = \{*\}$. The operational semantics of this process language is given as a labelled transition system $\mathcal{T} = (\mathcal{P}, Act_*, \to)$, where $\mathcal{P}$ is the set of closed process terms (the notions of open and closed terms are as usual) and $\to \subseteq \mathcal{P} \times Act_* \times \mathcal{P}$ is given as the least relation satisfying the rules of table 1. We shall refer to elements of $\mathcal{P}$ simply as *processes*.

The assertions of the modal μ-calculus will be given in a negation-free version and we use the construction of Winskel [Winskel, 1989] of *tagging* fixed points with sets of processes. Thus the assertions are constructed from the following grammar:

$$\begin{aligned} A ::= {} & A_0 \vee A_1 \mid A_0 \wedge A_1 \mid \langle\kappa\rangle A \mid [\kappa]A \mid \\ & X \mid \mu X\{U\}A \mid \nu X\{U\}A \end{aligned}$$

where $U \subseteq \mathcal{P}$ is a set of *tags* and X ranges over a set of assertion variables. The usual tag-free fixed points $\mu X.A$ and $\nu X.A$ are special cases corresponding to empty tag sets. We have chosen to let the modalities *diamond* $\langle\kappa\rangle A$ and *box* $[\kappa]A$ range over sets of composite actions $\kappa \subseteq Act_*$ instead of just the more commonly single actions.

The semantics of assertions $[\![A]\!]\rho \subseteq \mathcal{P}$ is given by induction on the structure of A; the map ρ is an *environment* taking all free variables of A to subsets of $\mathcal{P}$. For the fixed points we observe that the bodies, when considered as functions of X, are monotonic on the complete lattice $(Pow(\mathcal{P}), \subseteq)$ and then appeal to the Knaster-Tarski fixed-point theorem [Tarski, 1955] for supplying a minimum fixed point, denoted by μ, and a maximum fixed point, denoted by ν:

$$
\begin{array}{rcl}
[\![A_0 \vee A_1]\!]\rho & = & [\![A_0]\!]\rho \cup [\![A_1]\!]\rho \\
[\![A_0 \wedge A_1]\!]\rho & = & [\![A_0]\!]\rho \cap [\![A_1]\!]\rho \\
[\![\langle\kappa\rangle A]\!]\rho & = & \{p \in \mathcal{P} \mid \exists \alpha \in \kappa\ \exists p'.\ p \stackrel{\alpha}{\to} p' \ \&\ p' \in [\![A]\!]\rho\} \\
[\![[\kappa] A]\!]\rho & = & \{p \in \mathcal{P} \mid \forall \alpha \in \kappa\ \forall p'.\ p \stackrel{\alpha}{\to} p' \ \Rightarrow\ p' \in [\![A]\!]\rho\} \\
[\![X]\!]\rho & = & \rho(X) \\
[\![\mu X\{U\}A]\!]\rho & = & \mu V.([\![A]\!]\rho[V/X] \setminus U) \\
[\![\nu X\{U\}A]\!]\rho & = & \nu V.([\![A]\!]\rho[V/X] \cup U)
\end{array}
$$

Satisfaction between a process p and a closed assertion A is now defined by, $p \models A$, iff, $p \in [\![A]\!]\rho$ for all ρ. For future reference we define:

Definition 1 Let S_p be the set of *sub-term reachable states* of the process p. I.e. the least set of states closed under

$$
\begin{array}{rl}
(i) & p \in S_p, \\
(ii) & \text{if } q \in S_p \text{ and } q \stackrel{\alpha}{\to} q' \text{ then } q' \in S_p, \\
(iii) & \text{if } q \in S_p \text{ and } q' \text{ is a closed subterm of } q \text{ then } q' \in S_p.
\end{array}
$$

Let R_p, the *reachable states* of p, be the least subset of S_p closed under (i) and (ii). $\square$

It is not hard to prove that if all recursive terms in a process p are *regular* (i.e. the body is built entirely from $\mathbf{0}$, $+$, $a.$, x, and *rec*) then S_p is finite. A recursion *rec* $x.t$ is said to be *guarded* if any occurrence of x in t is inside a prefix.

3 The proof system

The proof system will be presented as "goal-oriented" proof rules defining inductively the relation $\vdash \subseteq \mathcal{P} \times ClAssn$ between processes and *closed assertions.* The rules naturally fall into three classes: Rules that do not involve the process operators, rules for the dynamic process operators $(\mathbf{0}, \mathbf{a}., +, rec)$, and finally rules for the static process operators $(\{\Xi\}, \lceil\Lambda, \times)$.

3.1 Rules for the fixed points, boolean connectives and idling modalities

The first class of rules, given in table 2, only depend on the structure of assertions. They encompass rules for the boolean connectives, modalities with the idling action and fixed points. These are straightforward rules that need little comment, except for the fixed-point rules. They are based on the following observation, originally due to Kozen, and later used as the key step in a local model checker by Winskel:

Lemma 1 (Reduction lemma) ([Kozen, 1983], [Winskel, 1989]) *For ψ a monotonic function on a powerset $Pow(D)$ with $p \in D$, we have*

$$p \in \mu V.\psi(V) \;\Leftrightarrow\; p \in \psi(\mu V.(\psi(V) \setminus \{p\})),$$

$$p \in \nu V.\psi(V) \;\Leftrightarrow\; p \in \psi(\nu V.(\psi(V) \cup \{p\})).$$

The last bi-implication holds for an arbitrary set P and inclusion instead of just for a singleton; the first not. Kozen [Kozen, 1983, Prop.5.7(vi)] proved the direction from right to left. Cleaveland [Cleaveland, 1990] used a quite similar lemma in showing completeness of his tableau method.

The right-hand sides of the bi-implications involve a slightly modified unfolding of the fixed points. For the minimum fixed point a single element, p, is removed in the unfolding; for the maximum it is added. The tagged fixed-point assertions were introduced to make this unfolding expressible directly in the logic. Thus under the assumption that $p \notin U$ the first bi-implication shows that $p \models \mu X\{U\}A$ if and only if $p \models A[\mu X\{U,p\}A/X]$, which shows soundness of the rule (μ). Similarly, the soundness of the maximum fixed point rule ($\nu 1$) follows from the second bi-implication.[1]

Remark We shall refer to the rules in the sequel by names constructed from the operators of the term and assertion that is involved in the rule. When this does not give a unique name we add numbers starting from 0. The names will be shown next to the rules in the tables. $\square$

[1] An alternative to the tags is to change the proof system into a *tableau system* where a similar effect is achieved by giving global success/failure criteria on the proof tree. See for example [Stirling and Walker, 1991] for an explanation of the relationship between the two approaches.

$$(\wedge)\ \frac{t \vdash A_0 \wedge A_1}{t \vdash A_0 \quad t \vdash A_1}$$

$$(\vee 0)\ \frac{t \vdash A_0 \vee A_1}{t \vdash A_0} \qquad (\vee 1)\ \frac{t \vdash A_0 \vee A_1}{t \vdash A_1}$$

$$([*])\ \frac{t \vdash [*, \kappa]A}{t \vdash A \quad t \vdash [\kappa]A}$$

$$(\langle * \rangle 0)\ \frac{t \vdash \langle *, \kappa \rangle A}{t \vdash A} \qquad (\langle * \rangle 1)\ \frac{t \vdash \langle *, \kappa \rangle A}{t \vdash \langle \kappa \rangle A}$$

$$(\mu)\ \frac{t \vdash \mu X\{U\}A}{t \vdash A[\mu X\{U, t\}A/X]}\ t \notin U$$

$$(\nu 0)\ \frac{t \vdash \nu X\{U, t\}A}{} \qquad (\nu 1)\ \frac{t \vdash \nu X\{U\}A}{t \vdash A[\nu X\{U, t\}A/X]}\ t \notin U$$

Table 2: Rules for the boolean connectives, idling modalities and fixed points.

3.2 Rules for the dynamic operators

What is missing now are rules for assertions where the top-level operator is a modality which do not involve an idling action. These remaining rules will depend on the structure of the process term, in different ways for the dynamic and the static operators. For the dynamic process operators they are rather direct consequences of the operational semantics, see table 3, once the following is observed for the recursion operator:

Proposition 1 *Assume rec $x.t$ is a closed process term, A a closed assertion, and κ a set of composite actions not containing $*$. Then*

$$\textit{rec } x.t \models [\kappa]A \;\Leftrightarrow\; t[\textit{rec } x.t/x] \models [\kappa]A,$$

$$\textit{rec } x.t \models \langle\kappa\rangle A \;\Leftrightarrow\; t[\textit{rec } x.t/x] \models \langle\kappa\rangle A.$$

Proof: Since there is only one operational rule for the recursion operator – the unfolding rule – *rec* $x.t$ and $t[\textit{rec } x.t/x]$ have syntactically the same α-successors for any $\alpha \neq *$. The proposition now follows from the observation

$(\mathbf{0}[])$	$\dfrac{\mathbf{0} \vdash [\kappa]A}{}$
$(.[]0)$	$\dfrac{a.t \vdash [a,\kappa]A}{t \vdash A}$
$(.[]1)$	$\dfrac{a.t \vdash [\kappa]A}{} \; a \notin \kappa$
$(.\langle\rangle)$	$\dfrac{a.t \vdash \langle a,\kappa\rangle A}{t \vdash A}$
$(+[])$	$\dfrac{t_0 + t_1 \vdash [\kappa]A}{t_0 \vdash [\kappa]A \quad t_1 \vdash [\kappa]A}$
$(+\langle\rangle 0)$	$\dfrac{t_0 + t_1 \vdash \langle\kappa\rangle A}{t_0 \vdash \langle\kappa\rangle A}$
$(+\langle\rangle 1)$	$\dfrac{t_0 + t_1 \vdash \langle\kappa\rangle A}{t_1 \vdash \langle\kappa\rangle A}$
$(\mathit{rec}[])$	$\dfrac{\mathit{rec}\ x.t \vdash [\kappa]A}{t[\mathit{rec}\ x.t/x] \vdash [\kappa]A}$
$(\mathit{rec}\langle\rangle)$	$\dfrac{\mathit{rec}\ x.t \vdash \langle\kappa\rangle A}{t[\mathit{rec}\ x.t/x] \vdash \langle\kappa\rangle A}$

Table 3: Dynamic process operators. All rules assume $* \notin \kappa$.

that for any closed process p, the judgements $p \models \langle\kappa\rangle A$ and $p \models [\kappa]A$ are fully determined by the κ-successors of p. □

It is important that the top-level assertion is a modality: The *successor states* of $\mathit{rec}\ x.t$ and its unfolded version are *syntactically identical* (since unfolding is the only operational rule for recursion), and thus satisfies the same set of assertions. But $\mathit{rec}\ x.t$ satisfies $\nu X\{\mathit{rec}\ x.t\}A$ whereas this is not necessarily the case for $t[\mathit{rec}\ x.t/x]$.

3.3 Rules for the static operators

In order to give rules for the static operators we shall extend the assertions with operators expressing the "pre-images" of the corresponding process operators. For relabelling, this mean that we allow assertions like $A\{\Xi\}$ with the semantic interpretation

$$[\![A\{\Xi\}]\!]\rho \;=\; \{p \mid p\{\Xi\} \in [\![A]\!]\rho\}.$$

Thus $t \models A\{\Xi\}$ if and only if $t\{\Xi\} \models A$. Hence, we include in the syntax these *extended assertions*:

$$A \;::=\; \ldots \mid A\{\Xi\} \mid A\restriction\Lambda \mid A/t$$

$(\{\}[])$	$\dfrac{t\{\Xi\} \vdash [\kappa]A}{t \vdash [\Xi^{-1}(\kappa)](A\{\Xi\})}$	$(\{\}\langle\rangle)$	$\dfrac{t\{\Xi\} \vdash \langle\kappa\rangle A}{t \vdash \langle\Xi^{-1}(\kappa)\rangle(A\{\Xi\})}$		
$(\restriction[])$	$\dfrac{t\restriction\Lambda \vdash [\kappa]A}{t \vdash [\Lambda \cap \kappa](A\restriction\Lambda)}$	$(\restriction\langle\rangle)$	$\dfrac{t\restriction\Lambda \vdash \langle\kappa\rangle A}{t \vdash \langle\Lambda \cap \kappa\rangle(A\restriction\Lambda)}$		
$(\{\})$	$\dfrac{t \vdash A\{\Xi\}}{t\{\Xi\} \vdash A}$	$(\restriction)$	$\dfrac{t \vdash A\restriction\Lambda}{t\restriction\Lambda \vdash A}$	$(\times)$	$\dfrac{t_0 \vdash A/t_1}{t_0 \times t_1 \vdash A}$

Table 4: Rules for eliminating relabelling and restriction from the process, and the three shift rules. The rules assume $* \notin \kappa$.

The semantic interpretations of the last two operators, restriction and *quotienting*, are:

$$\begin{array}{rcl} [\![A\restriction\Lambda]\!]\rho & = & \{p \mid p\restriction\Lambda \in [\![A]\!]\rho\} \\ [\![A/t]\!]\rho & = & \{p \mid p \times t \in [\![A]\!]\rho\} \end{array}$$

The new assertion operators will be used in giving rules for the modalities. For instance, one of the rules for relabelling will be

$$\frac{t\{\Xi\} \vdash [\kappa]A}{t \vdash [\Xi^{-1}(\kappa)](A\{\Xi\})}$$

Notice, that the operator $\{\Xi\}$ is applied to an assertion "guarded" by a box-modality. This box-modality can be removed by further application of the rules. At some point we might end up with $\{\Xi\}$ being applied at the top-level, and the rule we choose to give for such an assertion is a *shift rule* that shifts the operator back to the process, see table 4.

Various versions of parallel composition has traditionally posed the greatest difficulties in giving compositional rules. To get an idea of the difficulties, suppose we are confronted with the satisfaction problem $t_0 \times t_1 \vdash A$ and we want to decompose this to satisfaction problems for t_0 and t_1 without inspecting the structure of t_0 and t_1. If we think of $t_0 \times t_1$ as an element of the two-dimensional "plane", $\mathcal{P}\times\mathcal{P}$, the assertion A will be some two-dimensional "shape" in this plane. A decomposition of A could now be constructed by taking fragments A_0 and A_1 of the two axes, such that t_0 should satisfy A_0 and t_1 should satisfy A_1. However, for this to be a complete decomposition, valid for all t_0 and t_1, we would need to have A equal to the product of A_0

and A_1. This product would always be a "rectangle" – something which is certainly not true for arbitrary A. One way to get around this problem is to approximate A from the inside by a set of pairs of assertions (A_0^i, A_1^i) forming rectangles, the union of which forms exactly A. However, as Winskel argues in [Winskel, 1990b] the presence of fixed points can force this to be an *infinite* set; resulting in a poor decomposition.[2]

Fortunately, if we are slightly less ambitious and allow ourselves to inspect the structure of one of the two components, we can do better. In the suggested picture, this corresponds to the fact that if we fix a point on one of the axes, we can project to the other and get a subset of $\mathcal{P}$. The task of decomposition is now to find the assertion expressing this projection. As we shall see in section 6, if the component is finite-state, it is possible to directly compute the projected assertion. But in the rules we will be more general and impose no restrictions on finiteness; in fact, the rules will be local and for the dynamic operators follow very closely the rules of table 3. The main difference is that we are now considering a process t' in a 'context' $t \times \_$ which, however, play no active role in the rules; all the rules are guided solely by the structure of t'.

As before with the idling modalities, we shall need some rules that allow actions idling in the right component to be taken outside of the modalities. In order to state these rules we use the auxiliary operation κ/α of quotienting a set of actions with respect to a particular action. This operation is defined by $\kappa/\alpha = \{\beta \mid \beta \times \alpha \in \kappa\}$. We also use $\kappa \setminus \_ \times *$ for the set of actions $\alpha \times \beta \in \kappa$ for which β is not $*$. These rules are given as the first three rules of table 5. They are easily seen to be sound. The last eight rules of table 5 are the rules for the dynamic operators.

When the right component t' is headed by a static operator, we simplify the right component at the expense of the left. Let the operation $l_\times(A)$ reassociate every modality and every tag of the form $\_ \times (\_ \times \_)$ in A to the left. Then, we change the product $t \times (t_0 \times t_1)$ to $(t \times t_0) \times t_1$ and perform the corresponding rearrangement on A by replacing it by $l_\times(A)$. Analogously, when t' is a relabelling we will exploit that $t \times (t'\{\Xi\})$ is equivalent to $(t \times t')\{\mathrm{Id} \times \Xi\}$, where Id is the identity relabelling and the product of relabellings

[2]An example of a difficult assertion is the assertion $\mathcal{B}$ from [Andersen, 1993] expressing bisimilarity: $p \times q \models \mathcal{B}$, iff, p and q are strongly bisimilar. Hence, $\mathcal{B}$ forms a diagonal in the "plane". A decomposition would include a rectangle for each equivalence class.

$$(\times[*])\ \frac{t \times t' \vdash [\kappa]A}{t \vdash [\kappa/*](A/t') \quad t \times t' \vdash [\kappa \setminus \_ \times *]A}$$

$$(\times\langle*\rangle 0)\ \frac{t \times t' \vdash \langle\kappa\rangle A}{t \vdash \langle\kappa/*\rangle(A/t')} \qquad (\times\langle*\rangle 1)\ \frac{t \times t' \vdash \langle\kappa\rangle A}{t \times t' \vdash \langle\kappa \setminus \_ \times *\rangle A}$$

The rules below all assume $\kappa/* = \emptyset$

$$(\times\mathbf{0}[])\ \frac{t \times \mathbf{0} \vdash [\kappa]A}{}$$

$$(\times.[])\ \frac{t \times a.p \vdash [\kappa]A}{t \vdash [\kappa/a](A/p)} \qquad (\times.\langle\rangle)\ \frac{t \times a.p \vdash \langle\kappa\rangle A}{t \vdash \langle\kappa/a\rangle(A/p)}$$

$$(\times+[])\ \frac{t \times (t_0 + t_1) \vdash [\kappa]A}{t \times t_0 \vdash [\kappa]A \quad t \times t_1 \vdash [\kappa]A}$$

$$(\times+\langle\rangle 0)\ \frac{t \times (t_0 + t_1) \vdash \langle\kappa\rangle A}{t \times t_0 \vdash \langle\kappa\rangle A} \qquad (\times+\langle\rangle 1)\ \frac{t \times (t_0 + t_1) \vdash \langle\kappa\rangle A}{t \times t_1 \vdash \langle\kappa\rangle A}$$

$$(\times rec[])\ \frac{t \times rec\ x.t' \vdash [\kappa]A}{t \times t'[rec\ x.t'/x] \vdash [\kappa]A}$$

$$(\times rec\langle\rangle)\ \frac{t \times rec\ x.t' \vdash \langle\kappa\rangle A}{t \times t'[rec\ x.t'/x] \vdash \langle\kappa\rangle A}$$

Table 5: Product rules for idling and dynamic operators. We use the abbreviations $\kappa/\alpha = \{\beta \mid \beta \times \alpha \in \kappa\}$ and $\kappa \setminus \_ \times * = \{\alpha \times \beta \in \kappa \mid \beta \neq *\}$.

$$(\times\times)\quad \frac{t \times (t_0 \times t_1) \vdash A}{(t \times t_0) \times t_1 \vdash l_\times(A)}$$

$$(\times\{\})\quad \frac{t \times (t'\{\Xi\}) \vdash A}{(t \times t')\{\mathrm{Id} \times \Xi\} \vdash l_{\{\Xi\}}(A)}$$

$$(\times\restriction)\quad \frac{t \times (t' \restriction \Lambda) \vdash A}{(t \times t') \restriction (Act_* \times \Lambda) \vdash l_{\restriction\Lambda}(A)}$$

Table 6: Product rules for static operators.

$\Xi_0 \times \Xi_1$ is defined by

$$\Xi_0 \times \Xi_1(\alpha) = \begin{cases} \Xi_0(\alpha_0) \times \Xi_1(\alpha_1) & \text{if } \alpha = \alpha_0 \times \alpha_1 \\ \alpha & \text{otherwise.} \end{cases}$$

The corresponding change on an assertion A is to replace every tag of the form $\_ \times (\_\{\Xi\})$ by a tag $(\_ \times \_)\{\mathrm{Id} \times \Xi\}$ and removing the others. Let $l_{\{\Xi\}}(A)$ be the result of performing this operation on A.

Finally, for restriction we exploit the equivalence between $t_0 \times (t_1 \restriction \Lambda)$ and $(t_0 \times t_1) \restriction (Act_* \times \Lambda)$ using the operation $l_{\restriction\Lambda}(A)$ to change the tags of A from $\_ \times (\_ \restriction \Lambda)$ to $(\_ \times \_) \restriction (Act_* \times \Lambda)$ and removing the others. This gives rise to the three rules of table 6 for the static operators.

4 Soundness

The rules are sound for *arbitrary* processes and complete for a set of finite-state processes, i.e. processes with only guarded regular recursions.

Many of the rules are both upwards and downwards sound as captured by the following proposition:

Proposition 2 *For all processes t, t_0, t_1 and t_2, closed extended assertions A and sets of composite actions κ not containing the idling action, we have*

$$\begin{array}{rrcl}
i) & t\{\Xi\} \models [\kappa]A & \Leftrightarrow & t \models [\Xi^{-1}(\kappa)](A\{\Xi\}) \\
i') & t\{\Xi\} \models \langle\kappa\rangle A & \Leftrightarrow & t \models \langle\Xi^{-1}(\kappa)\rangle(A\{\Xi\}) \\
ii) & t \upharpoonright \Lambda \models [\kappa]A & \Leftrightarrow & t \models [\Lambda \cap \kappa](A \upharpoonright \Lambda) \\
ii') & t \upharpoonright \Lambda \models \langle\kappa\rangle A & \Leftrightarrow & t \models \langle\Lambda \cap \kappa\rangle(A \upharpoonright \Lambda) \\
iii) & t \models A\{\Xi\} & \Leftrightarrow & t\{\Xi\} \models A \\
iv) & t \models A \upharpoonright \Lambda & \Leftrightarrow & t \upharpoonright \Lambda \models A \\
v) & t_0 \models A/t_1 & \Leftrightarrow & t_0 \times t_1 \models A.
\end{array}$$

Proof: For the bi-implications *i*–*ii'* observe that there is only one rule that applies to a relabelled term and to a restricted term. Combining this with the semantics of extended assertions immediately results in the four bi-implications.

The bi-implications *iii*–*v* follow directly from the definition of the semantics of extended assertions. □

Before arguing for soundness of the product static rules, table 6, we need a simple lemma on the semantics of assertions and another lemma relating fixed points in different lattices.

Lemma 2 (Locality Lemma, [Andersen and Winskel, 1992]) *Assume that U is a set of processes closed under $\rightarrow$. Let $\mathcal{T}_U$ be the restriction of $\mathcal{T}$ to U. By $[\![A]\!]_U\rho$ we denote the semantics of A with respect to this restricted transition system: Tag-sets are restricted to U and fixed points are found in the complete lattice $Pow(U)$. Then for all A and ρ,*

$$[\![A]\!]\rho \cap U \;=\; [\![A]\!]_U \rho_U$$

where $\rho_U(X) = \rho(X) \cap U$.

To see how we shall employ this lemma, take $L = \{t_0 \times (t_1\{\Xi\}) \mid t_0, t_1 \in \mathcal{P}, \Xi \text{ a relabelling}\}$ and $R = \{(t_0 \times t_1)\{\mathrm{Id} \times \Xi\} \mid t_0, t_1 \in \mathcal{P}, \Xi \text{ a relabelling}\}$. Since no operational rules can remove the static operators, L and R are closed under $\rightarrow$. Now, by the locality lemma

$$t_0 \times (t_1\{\Xi\}) \models A \;\Leftrightarrow\; t_0 \times (t_1\{\Xi\}) \in [\![A]\!]_L \rho_L$$

for all ρ. Hence, we lose no information by taking the local view of the semantics. Similarly for processes in R we have

$$(t_0 \times t_1)\{\mathrm{Id} \times \Xi\} \models A \;\Leftrightarrow\; (t_0 \times t_1)\{\mathrm{Id} \times \Xi\} \in [\![A]\!]_R \rho_R.$$

Our proof of soundness of $(\times\{\})$ will relate these two local views of the semantics. In doing this we will need to relate fixed points in the two complete lattices $Pow(L)$ and $Pow(R)$. The following lemma gives the means to relate minimum fixed points – a dual version of it for relating maximum fixed points:

Lemma 3 *Let D and E be complete sub-lattices of $Pow(\mathcal{P})$. Let $in : D \to E$ be an ω-continuous function such that $in(\bot_D) = \bot_E$. Suppose $\phi : E \to E$ and $\theta : D \to D$ are monotonic functions such that*

$$in \circ \theta \;=\; \phi \circ in.$$

Then

$$in(\mu\theta) \;=\; \mu\phi.$$

See [Andersen and Winskel, 1992] or [Andersen, 1993] for a proof.

In proving $(\times\{\})$ sound we shall use the lemma in the case where D is $Pow(L)$, E is $Pow(R)$ and in is the ω-continuous function $l_{\{\Xi\}}$ mapping $t_0 \times (t_1\{\Xi\})$ to $(t_0 \times t_1)\{\mathrm{Id} \times \Xi\}$. Similar setups work for the rules $(\times\times)$ and $(\times\restriction)$:

Lemma 4 *For all processes t_0, t_1, t_2 and closed extended assertions A we have:*

$$\begin{array}{rrcl}
vi) & t_0 \times (t_1 \times t_2) \models A & \Leftrightarrow & (t_0 \times t_1) \times t_2 \models l_\times(A) \\
vii) & t_0 \times (t_1\{\Xi\}) \models A & \Leftrightarrow & (t_0 \times t_1)\{\mathrm{Id} \times \Xi\} \models l_{\{\Xi\}}(A) \\
viii) & t_0 \times (t_1 \restriction \Lambda) \models A & \Leftrightarrow & (t_0 \times t_1) \restriction (Act_* \times \Lambda) \models l_{\restriction\Lambda}(A).
\end{array}$$

Proof: We shall prove vii the remaining two bi-implications are similar. By structural induction on A we prove for all A, $P(A)$ where $P(A)$ is defined by, $P(A)$ holds if and only if

$$l_{\{\Xi\}}([\![A]\!]_L \rho) \;=\; [\![l_{\{\Xi\}}(A)]\!]_R l_{\{\Xi\}} \circ \rho$$

for all environments ρ assigning variables to subsets of L. Having proven $P(A)$ it follows immediately by the locality lemma that for any closed A, if $t_0 \times (t_1\{\Xi\}) \models A$ then $l_{\{\Xi\}}(t_0 \times (t_1\{\Xi\})) = (t_0 \times t_1)\{\mathrm{Id} \times \Xi\}$ satisfies $l_{\{\Xi\}}(A)$.

To prove $P(A)$ we consider the possible forms of A.

Case $A \equiv X$. This case follows directly from the semantic clause for X: $l_{\{\Xi\}}([\![X]\!]_L\rho) \; = \; l_{\{\Xi\}}(\rho(X))$.

Case $A \equiv \mu X\{U\}A$. Let $\theta(V) = [\![A]\!]_L\rho[V/X] \cup U$ and $\phi(V) = [\![A]\!]_R(l_{\{\Xi\}} \circ \rho)[V/X] \cup l_{\{\Xi\}}(U)$. By the induction hypothesis it follows that

$$l_{\{\Xi\}}(\theta(V)) = \phi(l_{\{\Xi\}}(V))$$

and hence, since both θ and ϕ are clearly monotonic, $l_{\{\Xi\}}(\emptyset) = \emptyset$ and $l_{\{\Xi\}}$ is ω-continuous it follows from lemma 3 that

$$l_{\{\Xi\}}(\mu\theta) = \mu\phi.$$

By expanding θ and ϕ and using the semantic clause for the minimum fixed points we get

$$l_{\{\Xi\}}([\![\mu X\{U\}A]\!]_L\rho) = [\![\mu X\{U\}A]\!]_R l_{\{\Xi\}} \circ \rho$$

completing the case for the minimum fixed point.

The case for the maximum fixed point is completely dual. The remaining cases for A are simple provided the following is observed for the modalities: A transition

$$t_0 \times (t_1\{\Xi\}) \stackrel{\alpha\times\beta}{\rightarrow} t_0' \times (t_1'\{\Xi\})$$

is possible, if and only if, $t_0 \stackrel{\alpha}{\rightarrow} t_0'$ and there exists a γ such that $t_1 \stackrel{\gamma}{\rightarrow} t_1'$ and $\Xi(\gamma) = \beta$. This will be the case, if and only if,

$$(t_0 \times t_1)\{\mathrm{Id} \times \Xi\} \stackrel{\alpha\times\beta}{\rightarrow} (t_0' \times t_1')\{\mathrm{Id} \times \Xi\}.$$

Hence, the two terms will always be capable of performing the same initial transitions. (In fact, they are strongly bisimilar cf. [Milner, 1989]). $\square$

We can now prove soundness for the full set of rules:

Theorem 1 (Soundness) *Assume a process t and a closed assertion A. If $t \vdash A$ can be proven using the rules of table 2, 3, 4, 5, and 6 then $t \models A$.*

Proof: As usual this is shown by proving that each rule is sound. We shall argue that they are sound even for extended assertions. Rules ($\wedge$), ($\vee$0) and ($\vee$1) are straightforward. The idling rules ($[]*$), ($\langle\rangle*0$) and ($\langle\rangle*1$) are straightforward since all processes t – due to the restrictions on Ξ and Λ – have the unique idling transition $t \xrightarrow{*} t$.

Soundness of the fixed point rules (μ), ($\nu 1$) follows from the reduction lemma and they are both upwards and downwards sound. Soundness of ($\nu 0$) is immediate from the semantics.

Soundness of ($\mathbf{0}[]$), ($.[]0$), ($.[]1$), ($.\langle\rangle$), ($+[]$), ($+\langle\rangle 0$) and ($+\langle\rangle 1$) can be proven directly from the operational semantics. The recursive process rules are both upwards and downwards sound as shown in proposition 1.

Soundness of the rules ($\{\Xi\}[]$), ($\{\Xi\}\langle\rangle$), ($\lceil\Lambda[]$), ($\lceil\Lambda\langle\rangle$), ($\{\Xi\}$), ($\lceil\Lambda$), and ($\times$) of table 4 follows from proposition 2. The idling rules ($\times[]*$), ($\times\langle\rangle*0$) and ($\times\langle\rangle*1$) follows directly from the operational semantics as for the$\times$-free versions. Similarly, the dynamic rules ($\times\mathbf{0}$), ($\times.[]$), ($\times.\langle\rangle$) etc. are proven like their $\times$-free counterparts ($\mathbf{0}$), ($.[]0$)/($.[]1$), ($.\langle\rangle$) etc.

Soundness of the product static rules ($\times\times$), ($\times\{\Xi\}$) and ($\times\lceil\Lambda$) follows also from proposition 2. $\square$

Notice, that the fixed-point rules, the dynamic process operator rules, and all the static operator rules – including the product rules – are both upwards and downwards sound. In fact, only the disjunction rules and the 8 diamond rules ($\langle*\rangle 0$), ($\langle*\rangle 1$), ($+\langle\rangle 0$), ($+\langle\rangle 1$), ($\times\langle*\rangle 0$), ($\times\langle*\rangle 1$), ($\times+\langle\rangle 0$), ($\times+\langle\rangle 1$) fail to be downwards sound.

5 Completeness

Central in our proof of completeness will be a well-founded relation on assertions:

Lemma 5 *The relation $\prec$ defined on closed assertions with tags from a finite set S by*

$$\begin{aligned} A \prec A' \text{ iff } & A \text{ is a proper subassertion of } A', \text{ or} \\ & A' \equiv \sigma X\{U\}B \text{ and} \\ & A \equiv B[\sigma X\{U,t\}B/X] \text{ for some } t \notin U, \end{aligned}$$

where σ is one of μ and ν, is well-founded.

The relation $\prec$ embodies the fact that the small modifications to the tags when unfolding the fixed points is enough to ensure that the fixed-point rules can only be applied a finite number of times before $t \in U$. It captures in a very precise manner the reason for termination of model checking algorithms based on the fixed-point rules (μ), ($\nu 0$) and ($\nu 1$) as in the works of Stirling and Walker [Stirling and Walker, 1991], Cleaveland [Cleaveland, 1990] and Winskel [Winskel, 1989].

The proof strategy in proving completeness is as follows. Assume a process p with a finite set of sub-term reachable states S_p. By well-founded induction using $\prec$ we show that for all $t \in S_p$, if $t \models A$ then $t \vdash A$. When A is of the form $[\kappa]B$ or $\langle\kappa\rangle B$ this will involve inspecting the structure of the term t. Thus we shall show by another induction, this time on t, how to construct from proofs of some $t_1 \vdash B, \ldots, t_n \vdash B$ where t_i is less than t and for all i $t_i \models B$, a proof of $t \vdash A$. The "less than" ordering we use on terms is based on a measure $w(t)$ that is roughly "the maximal depth to a prefix, nil or variable in t," which, however, gives more weight to the second component of a product than to the first. Hence, simplifying the second component at the expense of the first, as it is done in the static rules, is still considered a way of making progress.

Theorem 2 (Completeness for finite-state processes) *If p is a process with guarded regular recursions then, for all closed assertions A with tags in S_p, if $p \models A$ then $p \vdash A$.*

Proofs of this theorem and lemma 5 can be found in the appendix.

To show an example of the usage of the rules, we will consider the CCS parallel composition $|$ as an abbreviation for $(\_ \times \_) \restriction \Lambda\{\Xi\}$ where Λ and Ξ are as follows. First, the actions Act are supposed to include a distinguished *internal action* τ and the remaining actions are called *names.* Associated with each name a is a co-name $\bar{a}$; such that $\bar{}$ forms a bijection on $Act \setminus \tau$. Then, take $\Lambda = \{a \times \bar{a}, a' \times *, * \times a' \mid a \in Act \setminus \tau, a' \in Act\}$, and let $\Xi(a \times \bar{a}) = \tau, \Xi(a' \times *) = \Xi(* \times a') = a'$ and on other actions α, $\Xi(\alpha) = \alpha$. It is not hard to see that $(p \times q) \restriction \Lambda\{\Xi\}$ will behave exactly as $p|q$.

Example This example illustrates how the compositionality facilitates proving a property about a process that contains infinite-state components – when the infinite-state behaviour is irrelevant for the property: Assume p and $q \equiv rec\ x.\tau.x + t$ are infinite-state processes (x might be free in t). We shall consider the process $p|q$ and prove that it has an infinite τ-loop as expressed by the assertion $\nu X\{\}\langle\tau\rangle X$.

$$
\begin{array}{cl}
p|q \vdash \nu X\{\}\langle\tau\rangle X & \\ \hline
p|q \vdash \langle\tau\rangle \nu X\{p|q\}\langle\tau\rangle X & (\nu 1) \\ \hline
(p \times q)\upharpoonright\Lambda \vdash \langle \Xi^{-1}(\tau)\rangle(\nu X\{p|q\}\langle\tau\rangle X)\{\Xi\} & (\{\}\langle\rangle) \\ \hline
p \times q \vdash \langle\kappa\rangle(\nu X\{p|q\}\langle\tau\rangle X)\{\Xi\}\upharpoonright\Lambda & (\upharpoonright\langle\rangle) \\ \hline
p \times (\tau.q + t[q/x]) \vdash \langle\kappa\rangle(\nu X\{p|q\}\langle\tau\rangle X)\{\Xi\}\upharpoonright\Lambda & (\times \mathit{rec}\langle\rangle) \\ \hline
p \times \tau.q \vdash \langle\kappa\rangle(\nu X\{p|q\}\langle\tau\rangle X)\{\Xi\}\upharpoonright\Lambda & (\times + \langle\rangle 0) \\ \hline
p \times \tau.q \vdash \langle\kappa \setminus \_ \times *\rangle(\nu X\{p|q\}\langle\tau\rangle X)\{\Xi\}\upharpoonright\Lambda & (\times\langle *\rangle 1) \\ \hline
p \vdash \langle *\rangle(\nu X\{p|q\}\langle\tau\rangle X)\{\Xi\}\upharpoonright\Lambda/q & (\times.\langle\rangle) \\ \hline
p \vdash (\nu X\{p|q\}\langle\tau\rangle X)\{\Xi\}\upharpoonright\Lambda/q & (\langle *\rangle 0) \\ \hline
p|q \vdash \nu X\{p|q\}\langle\tau\rangle X & (\times), (\upharpoonright), (\{\}) \\ \hline
 & (\nu 0)
\end{array}
$$

Figure 1: A proof tree for the example.

Let $\kappa = \Lambda \cap \Xi^{-1}(\tau) = \{a \times \bar{a} \mid a \in \mathit{Act} \setminus \tau\} \cup \{\tau \times *, * \times \tau\}$. The proof tree is given in figure 1. Note that in the application of rule $(\times.\langle\rangle)$, we are using $(\kappa \setminus \_ \times *)/\tau = \{*\}$.

6 Reductions

There is an alternative approach to compositionality, followed in [Andersen and Winskel, 1992] and to some extent in [Larsen and Xinxin, 1990], based on the idea of *reductions*. A reduction transforms a satisfaction problem for a composite process $op(t_1, \ldots, t_n) \vdash A$ into a *boolean expression* over satisfaction problems $t_1 \vdash A_1, \ldots, t_n \vdash A_n$ for the subterms of the process – independent of the structure of these. Simple examples of reductions can be derived from:

$$
\begin{aligned}
t_0 + t_1 \models [\kappa]A &\Leftrightarrow (t_0 \models [\kappa]A) \text{ and } (t_1 \models [\kappa]A),\\
t_0 + t_1 \models \langle\kappa\rangle A &\Leftrightarrow (t_0 \models \langle\kappa\rangle A) \text{ or } (t_1 \models \langle\kappa\rangle A).
\end{aligned}
$$

In general, the reductions will be more involved. However, for the relabelling and restriction it is possible to give quite concise reductions. They simply change the modalities (and the tags) of the assertion and leave everything else unchanged. In the context of our proof rules such a reduction can be seen as a means for eliminating the extended assertions. I.e. for any assertion A, equivalent assertions $e(A\{\Xi\})$ and $e(A\restriction\Lambda)$ with $\{\Xi\}$ and $\restriction\Lambda$ removed, can be found. Table 7 shows these reductions. An alternative to the rules ($\{\Xi\}[]$) and ($\{\Xi\}\langle\rangle$) could now be

$$\frac{t\{\Xi\} \vdash A}{t \vdash e(A\{\Xi\})}$$

Thus, no extended assertion will be introduced by this new rule.

$$
\begin{aligned}
e(X\{\Xi\}) &= X \\
e(A_0 \wedge A_1\{\Xi\}) &= e(A_0\{\Xi\}) \wedge e(A_1\{\Xi\}) \\
e(A_0 \vee A_1\{\Xi\}) &= e(A_0\{\Xi\}) \vee e(A_1\{\Xi\}) \\
e([\kappa]A\{\Xi\}) &= [\Xi^{-1}(\kappa)]e(A\{\Xi\}) \\
e(\langle\kappa\rangle A\{\Xi\}) &= \langle\Xi^{-1}(\kappa)\rangle e(A\{\Xi\}) \\
e(\nu X\{U\}A\{\Xi\}) &= \nu X\{U\{\Xi\}\}e(A\{\Xi\}) \\
e(\mu X\{U\}A\{\Xi\}) &= \mu X\{U\{\Xi\}\}e(A\{\Xi\}) \\
\\
e(X\restriction\Lambda) &= X \\
e(A_0 \wedge A_1\restriction\Lambda) &= e(A_0\restriction\Lambda) \wedge e(A_1\restriction\Lambda) \\
e(A_0 \vee A_1\restriction\Lambda) &= e(A_0\restriction\Lambda) \vee e(A_1\restriction\Lambda) \\
e([\kappa]A\restriction\Lambda) &= [\Lambda \cap \kappa]e(A\restriction\Lambda) \\
e(\langle\kappa\rangle A\restriction\Lambda) &= \langle\Lambda \cap \kappa\rangle e(A\restriction\Lambda) \\
e(\nu X\{U\}A\restriction\Lambda) &= \nu X\{U\restriction\Lambda\}e(A\restriction\Lambda) \\
e(\mu X\{U\}A\restriction\Lambda) &= \mu X\{U\restriction\Lambda\}e(A\restriction\Lambda)
\end{aligned}
$$

Table 7: Reductions for relabelling and restriction. Recall, $U\{\Xi\} = \{p \mid p\{\Xi\} \in U\}$ and $U\restriction\Lambda = \{p \mid p\restriction\Lambda \in U\}$.

If t is a finite-state process, also the quotienting A/t can be removed by a reduction. To give this reduction we need to introduce tagged *simultaneous* fixed points. Let σ be any one of μ and ν. Then the syntax is:

$$\sigma X_1\{U_1\}\ldots X_n\{U_n\}(A_1,\ldots,A_n) \downarrow X_i,$$

abbreviated as $\sigma\vec{X}\{\vec{U}\}\vec{A} \downarrow X_i$. The semantics should be clear. The reduction is given in table 8. An alternative rule for product could now be

$$\frac{t_0 \times t_1 \vdash A}{t_0 \vdash e(A/t_1)},$$

$$
\begin{array}{|rl|}
\hline
e(X/p) = & X_p \\
e(A_0 \vee A_1/p) = & e(A_0/p) \vee e(A_1/p) \\
e(A_0 \wedge A_1/p) = & e(A_0/p) \wedge e(A_1/p) \\
e(\langle\kappa\rangle A/p) = & \bigvee\{\langle\alpha\rangle e(A/p') \mid \exists\alpha\times\beta\in\kappa.\ p \xrightarrow{\beta} p'\} \\
e([\kappa]A/p) = & \bigwedge\{[\alpha]e(A/p') \mid \exists\alpha\times\beta\in\kappa.\ p \xrightarrow{\beta} p'\} \\
e(\sigma X\{U\}A/p) = & \sigma X_{p_1}\{U/p_1\}\cdots X_{p_n}\{U/p_n\}.(e(A/p_1),\ldots,e(A/p_n)) \downarrow X_p \\
& \text{where } \{p_1,\ldots,p_n\} = R_p \\
\hline
\end{array}
$$

Table 8: Reduction for quotienting. Recall, $U/p = \{t \mid t \times p \in U\}$.

which, again, does not introduce any extended assertion. The price is, that the new rule is only applicable for finite-state processes, and we must now consider simultaneous fixed points. The simultaneous fixed points can be converted into simple fixed points using the Scott-Bekič principle [Bekič, 1984], thereby potentially increasing the size of the assertion considerably. A more appealing approach would be to extend the fixed-point rules to simultaneous fixed points. Then, for example, (μ) should be replaced by

$$\frac{t \vdash \mu\vec{X}\{\vec{U}\}\vec{A} \downarrow X_i}{t \vdash A_i[\mu\vec{X}\{\vec{U'}\}\vec{A}/\vec{X}]},$$

where $\vec{U'} = (U_1,\ldots,U_{i-1},U \cup \{t\},U_{i+1},\ldots,U_n)$ and the substitution $[\mu\vec{X}\{\vec{U'}\}\vec{A}/\vec{X}]$ is an abbreviation for $[\mu\vec{X}\{\vec{U'}\}\vec{A} \downarrow X_1/X_1,\ldots,\mu\vec{X}\{\vec{U'}\}\vec{A} \downarrow X_n/X_n]$.

(Proving the above reductions correct is an easy generalisation to tagged fixed points of the proofs in [Andersen and Winskel, 1992] and [Andersen, 1993].)

7 Conclusion

The idea of compositionality being "not looking into the structure of subprocesses" could be formalised using a set of "meta-variables" $\hat{x},\hat{y},\ldots$ distinct from the recursion variables. We should think of a variable $\hat{x}$ as being a yet undefined process – a "hole" in the term. Any proof carried out with such variables appearing in the terms, would then be valid for *all* instantiations of the variable – capturing the reusability of proofs. However, in defining the substitution on terms with meta-variables, a little care must be taken. In, for example, $\mathit{rec}\ x.a.\hat{y}$ we have the undefined process $\hat{y}$, which we might at some point decide to instantiate to the term x. Thus we would require $(\mathit{rec}\ x.a.\hat{y})[x/\hat{y}] = \mathit{rec}\ x.a.x$. (Also, a substitution like $\hat{y}[\mathit{rec}\ x.a.\hat{y}/x]$ cannot be reduced.)

It is interesting that the rules for recursion in combination with the tagging could actually help us in finding appropriate instantiations of meta-variables. Consider as an example the term $\mathit{rec}\ x.a.\hat{y}$ and the assertion $\nu X\{\}\langle a\rangle X$ expressing the existence of an infinite a-path. Using, in sequence, the rules $(\nu 1)$,$(\mathit{rec}\langle\rangle)$,$(.\langle\rangle)$ we will end up with

$$\hat{y}[\mathit{rec}\ x.a.\hat{y}/x] \vdash \nu X\{\mathit{rec}\ x.a.\hat{y}\}\langle a\rangle X.$$

Suppose we would try to apply rule $(\nu 0)$ in proving this valid. Then we would have to solve the equation $\hat{y}[\mathit{rec}\ x.a.\hat{y}/x] = \mathit{rec}\ x.a.\hat{y}$. A solution is to substitute x for $\hat{y}$, arriving at $\mathit{rec}\ x.a.x \vdash \nu X\{\mathit{rec}\ x.a.x\}\langle a\rangle X$, which by rule $(\nu 0)$ is valid.[3]

Returning to the proof system, we notice that compared to the earlier work of Stirling, Winskel, and Andersen and Winskel, the rules are few and quite simple. In particular, only three simple rules are needed to deal with fixed-point assertions, two to deal with recursive processes.

A useful amendment to the system is the possibility of relaxing the condition in $(\nu 0)$ that t should be an element of the tags of the maximum fixed-point to simply be strongly bisimilar to one of the tags. This amendment is straightforward since satisfaction in the modal μ-calculus is invariant under strong bisimulation, provided the tags are interpreted as equivalences classes. Another useful amendment would then be to combine the proof system with a proof system for bisimulation equivalence on processes.

In a recent paper [Andersen, 1995] it is shown how the quotienting combined with heuristics for simplifying simultaneous fixed points gives rise to a technique for handling the state explosion problem.

[3]The reduction for recursion given in [Andersen and Winskel, 1992] would, using some simplification steps, transform the satisfaction problem $\mathit{rec}\ x.a.\hat{y} \vdash \nu X.\langle a\rangle X$ into the problem $\hat{y} \vdash \nu X.(\langle a\rangle X \vee \{x\})$, where $\{x\}$ is an assertion true at the variable x – called a state identifier there. Thus it can immediately be seen that substituting x for $\hat{y}$ yields a solution. That reduction, however, is rather more involved and does not seem to give rise easily to a corresponding proof rule.

Appendix. Proofs

This appendix contains proofs of lemma 5 and theorem 2.

Lemma 5 *The relation $\prec$ defined on closed assertions with tags from a finite set S by*

$$\begin{array}{ll} A \prec A' \textit{ iff } & A \textit{ is a proper subassertion of } A', \textit{ or} \\ & A' \equiv \sigma X\{U\}B \textit{ and} \\ & A \equiv B[\sigma X\{U,t\}B/X] \textit{ for some } t \notin U, \end{array}$$

where σ is one of μ and ν, is well-founded.

Proof: Take the predicate $Q(A)$ on closed assertions A with tags in S to be defined by

$$Q(A) \;\Leftrightarrow_{\text{def}}\; \text{all } \prec\text{-decreasing sequences from } A \text{ are finite.}$$

Extend this to open terms by

$$\begin{array}{ll} Q^+(A) \;\Leftrightarrow_{\text{def}} & \forall\theta : FV(A) \to \mathit{ClAssn}. \\ & (\forall X \in FV(A).Q(\theta(X))) \;\Rightarrow\; Q(A[\theta]). \end{array}$$

Observe that if A is closed $Q^+(A)$ is simply $Q(A)$. The proof is by well-founded induction on a slightly different relation $\prec'$ defined by

$$\begin{array}{ll} A' \prec' A \text{ iff } & A' \text{ is a proper subassertion of } A, \text{ or} \\ & A \equiv \sigma X\{U\}B \text{ and} \\ & A' \equiv \sigma X\{U,t\}B \text{ for some } t \notin U. \end{array}$$

Since tags belong to the finite set S this relation is easily seen to be well-founded. Thus assume for all $A' \prec' A$, $Q^+(A)$ holds and $\forall X \in FV(A).Q(\theta(X))$. We consider the possible first successor A' in a $\prec$-decreasing sequence $A[\theta] \succ A'$ and argue that any continuation of the sequence must be finite. We consider the two possible reasons for $A[\theta] \succ A'$.

Case 1. A' is a proper subassertion of $A[\theta]$. Then either there exists a subassertion A'' of A such that $A''[\theta] \equiv A'$, or A' is a subassertion of some $\theta(X)$. In the first case the result follows from the induction hypothesis since $A'' \prec' A$; in the second it follows immediately from the assumption $Q(\theta(X))$.

Case 2. In this case, $A' \equiv B[\sigma X\{U,t\}B/X]$ and $A[\theta] \equiv \sigma X\{U\}B$. Either $A \equiv Y$ and $\theta(Y) = \sigma X\{U\}B$ or $A \equiv \sigma X\{U\}(B'[\theta])$ for some B'. In the first case the result follows from the assumption of $Q(\theta(Y))$; in the second it can be shown from the induction hypothesis as follows. Since $B \equiv B'[\theta]$ and $X \notin FV(A)$, we can write A' as

$$B'[\theta][\sigma X\{U,t\}(B'[\theta])/X] \equiv B'[\sigma X\{U,t\}B'/X][\theta].$$

Hence, since $\sigma X\{U,t\}B' \prec' \sigma X\{U\}B'$ it follows from the induction hypothesis that $Q^+(\sigma X\{U,t\}B')$ holds.

Take $\theta'(Y) = \theta(Y)$ for $Y \neq X$ and $\theta'(X) = \sigma X\{U,t\}(B'[\theta])$. Thus we have just argued $Q(\theta'(X))$ and surely $Q(\theta'(Y))$ for all $Y \neq X$. Since B' is a subassertion of A and therefore $B' \prec' A$ we can again use the induction hypothesis to conclude $Q(A[\theta])$.

□

Let the measure $w(t)$ be defined by structural induction on terms t by

$$\begin{aligned}
w(\mathbf{0}) = w(x) &= 0 \\
w(a.t) &= 0 \\
w(t_0 + t_1) &= 1 + \max\{w(t_0), w(t_1)\} \\
w(rec\ x.t) &= 1 + w(t) \\
w(t\{\Xi\}) = w(t \restriction \Lambda) &= 1 + w(t) \\
w(t_0 \times t_1) &= 1 + w(t_0) + 2w(t_1).
\end{aligned}$$

We can now prove the following lemma:

Lemma 3 *Assume a closed assertion B and a closed term t with guarded, regular recursions. If $t \models [\kappa]B$ ($t \models \langle\kappa\rangle B$) then there exists some $t_1, \ldots, t_n$ with $t_i \models B$ and from $t_1 \vdash B, \ldots, t_n \vdash B$ there is a proof of $t \vdash [\kappa]B$ ($t \vdash \langle\kappa\rangle B$).*

Proof: We prove the claim by showing $\forall t.P(t)$ using well-founded induction on t with the ordering induced by $w(t)$ where

$$\begin{aligned}
P(t) \Leftrightarrow_{\text{def}}\ & \text{for all closed, extended assertions } A, \\
& \text{if } t \models [\kappa]A \text{ then} \\
& \quad \exists t_1, \ldots, t_n.\ t_i \models A, \text{ and} \\
& \quad t \vdash [\kappa]A \text{ can be proven from } \{t_i \vdash A\}_i.
\end{aligned}$$

We shall only consider the case for the box-modality, the case of diamond-modality is similar. Thus assume for all t' with $w(t') < w(t)$ that $P(t')$ holds and assume further that $t \models [\kappa]A$. We shall establish $P(t)$ on these assumptions by considering the possible forms of t.

However, consider first the case where $* \in \kappa$. Then from the semantics we observe that $t \models A$ and $t \models [\kappa \setminus *]A$. The first is already on the required form hence take $t_1 \equiv t$; for $t \models [\kappa \setminus *]A$ the steps below assuming $* \notin \kappa$ provides the required remaining $t_2, \ldots, t_n$ to establish $P(t)$ using rule $([]*)$. Thus assume in the sequel $* \notin \kappa$.

$t \equiv \mathbf{0}$. Immediate from rule $(\mathbf{0}[])$.

$t \equiv a.t'$. If $a \in \kappa$ then $t' \models A$ and rule $(.[]0)$ gives a proof of $a.t' \vdash [\kappa]A$ from a proof of $t' \vdash A$. This shows $P(t)$ in this case.

If $a \notin \kappa$ then rule $(.[]1)$ immediately gives a proof of $a.t' \vdash [\kappa]A$ showing $P(t)$ in this case.

$t \equiv t_0 + t_1$. It follows from the semantics of assertions that $t_0 \models [\kappa]A$ and $t_1 \models [\kappa]A$, hence since $w(t_0) < w(t)$ and $w(t_1) < w(t)$ if follows by induction that there exists $t_0^1, \ldots, t_0^m$ and $t_1^1, \ldots, t_1^n$ with $t_0^i \models A$ and $t_1^j \models A$ such that proofs of $t_0 \vdash A$ and $t_1 \vdash A$ can be constructed from proofs of $t_0^i \vdash A$ and $t_1^j \vdash A$. Thus using rule $(+[])$ we can get a proof of $t_0 + t_1 \vdash A$ completing this case.

$t \equiv \mathit{rec}\ x.t'$. It follows from proposition 1 that $t'[\mathit{rec}\ x.t'/x] \models [\kappa]A$. Now, since all recursions are guarded and regular $w(t'[\mathit{rec}\ x.t'/x]) < w(\mathit{rec}\ x.t')$ hence by the induction hypothesis there exists $t_1 \models A, \ldots, t_n \models A$ such that a proof of $t'[\mathit{rec}\ x.t'/x] \vdash [\kappa]A$ can be constructed from proofs of $t_i \vdash A$. Applying rule $(\mathit{rec}[])$ to such a proof we have shown $P(t)$ in this case.

$t \equiv t'\{\Xi\}$. It follows from downwards soundness of rule $(\{\Xi\}[])$ that $t' \models [\Xi^{-1}(\kappa)](A\{\Xi\})$. Since $w(t') < w(t)$ it follows by induction that there exists $t'_1, \ldots, t'_n$ such that $t'_i \models A\{\Xi\}$ and that from proofs of $t'_i \vdash A\{\Xi\}$ we can construct a proof of $t' \vdash [\Xi^{-1}(\kappa)](A\{\Xi\})$. Now, to extend this to a proof of $t'\{\Xi\} \vdash [\kappa]A$ first take $t_i \equiv t'_i\{\Xi\}$. Hence from proofs of $t_i \vdash A$, i.e. $t'_i\{\Xi\} \vdash A$, we get proofs of $t'_i \vdash A\{\Xi\}$ using rule $(\{\Xi\})$. Finally, using rule $(\{\Xi\}[])$ we get a proof of $t'\{\Xi\} \vdash [\kappa]A$ from a proof of $t' \vdash [\Xi^{-1}(\kappa)](A\{\Xi\})$ which as we have just argued can be proven from $t_1 \vdash A, \ldots, t_n \vdash A$.

$t \equiv t' \restriction \Lambda$. As above but using rules $(\restriction \Lambda)$ and $(\restriction \Lambda[])$.

$t \equiv t_0 \times t_1$.

If $\kappa/* \neq \emptyset$ we can remove the set $(\kappa/*) \times \{*\}$ by applying rule $(\times[]*)$ and proceed as below – exactly like in the case of $* \in \kappa$ considered in the beginning of the proof. Hence, in the sequel assume $\kappa/* = \emptyset$ and consider the possible forms of t_1.

$t_1 \equiv \mathbf{0}$, $t_1 \equiv a.t'$, $t_1 \equiv t_1' + t_1''$, $t_1 \equiv \mathit{rec}\ x.t'$. Analogous to the cases above. See the discussion in section 3.3 about the relationship between the product dynamic rules and the dynamic rules.

$t_1 \equiv t_1' \times t_1''$. A little bit of arithmetic shows $w((t_0 \times t_1') \times t_1'') < w(t)$:

$$\begin{array}{ll} \multicolumn{2}{l}{w((t_0 \times t_1') \times t_1'')} \\ = & 1 + w(t_0 \times t_1') + 2w(t_1'') \\ = & 1 + (1 + w(t_0) + 2w(t_1')) + 2w(t_1'') \\ = & 1 + w(t_0) + 1 + 2w(t_1') + 2w(t_1'') \\ < & 1 + w(t_0) + 2(1 + w(t_1') + 2w(t_1'')) \\ = & w(t_0 \times (t_1' \times t_1'')) \ = \ w(t) \end{array}$$

Thus $P(t)$ follows from the induction hypothesis and rule $(\times\times)$.

$t_1 \equiv t_1'\{\Xi\}$. As above we compute:

$$\begin{array}{ll} \multicolumn{2}{l}{w((t_0 \times t_1)\{\Xi\})} \\ = & 1 + w(t_0 \times t_1) \\ = & 1 + (1 + w(t_0) + 2w(t_1)) \\ < & 1 + w(t_0) + 2(1 + w(t_1)) \\ = & w(t_0 \times (t_1\{\Xi\})) \end{array}$$

Thus $P(t)$ follows from the induction hypothesis and rule $(\times\{\Xi\})$.

$t_1 \equiv t_1' \restriction \Lambda$. As above.

$\square$

The proof of completeness now follows by well-founded induction on the relation $\prec$:

Theorem 2 (Completeness for finite-state processes) *If p is a process with guarded regular recursions then for all closed assertions A with tags in S_p,*

$$p \models A \Rightarrow p \vdash A.$$

Proof: Let $Q(A)$ be defined on closed assertions with tags in S_p by

$$Q(A) \Leftrightarrow_{\mathrm{def}} \forall t \in S_p.\ t \models A \ \Rightarrow\ t \vdash A.$$

We prove $Q(A)$ for all closed assertions with tags in S_p by induction on $\prec$. Hence assume $Q(A')$ for all $A' \prec A$.

We consider the potential forms of A.

$A \equiv X$. Impossible since A is assumed to be closed.

$A \equiv A_0 \wedge A_1$. Since $t \models A_0 \wedge A_1$ implies $t \models A_0$ and $t \models A_1$, and, moreover, $A_0 \prec A$, and $A_1 \prec A$ the result follows from the induction hypothesis applying rule $(\wedge)$.

$A \equiv A_0 \vee A_1$. Since $t \models A_0 \vee A_1$ implies $t \models A_0$ or $t \models A_1$, and, moreover, $A_0 \prec A$, and $A_1 \prec A$ the result follows from the induction hypothesis applying either rule $(\vee 0)$ or $(\vee 1)$.

$A \equiv \mu X\{U\}B$. From lemma 1 it follows that if $t \models \mu X\{U\}B$ then $t \models B[\mu X\{U,t\}B/X]$ and as it can easily be seen from the semantics of tagged minimum fixed points, $t \notin U$. Thus rule (μ) can be applied to yield a proof of $t \vdash \mu X\{U\}B$ from a proof of $t \vdash B[\mu X\{U,t\}B/X]$. Since $B[\mu X\{U,t\}B/X] \prec \mu X\{U\}B$ we have by the induction hypothesis a proof of $B[\mu X\{U,t\}B/X]$ completing this case.

$A \equiv \nu X\{U\}B$. If $t \in U$, rule $(\nu 0)$ immediately yields a proof of $t \vdash \nu X\{U\}B$. If $t \notin U$ but $t \models \nu X\{U\}B$ if follows from lemma 1 that $t \models B[\nu X\{U,t\}B/X]$ thus rule $(\nu 1)$ gives a proof of $t \vdash \nu X\{U\}B$ from a proof of $t \vdash B[\nu X\{U,t\}B/X]$. Since $B[\nu X\{U,t\}B/X] \prec \nu X\{U\}B$ we have by the induction hypothesis a proof of $B[\mu X\{U,t\}B/X]$ completing this case.

$A \equiv [\kappa]B,\ A \equiv \langle\kappa\rangle B$. Assuming $t \models [\kappa]B$ it follows from lemma 3 that there exists $t_1, \ldots, t_n$ such that $t_i \models B$ and $t \models [\kappa]B$ can be proven from proofs of $t_i \vdash B$. However, since $B \prec [\kappa]B$ it follows from the induction hypothesis that such proofs do indeed exist, completing the case for the box-modality. The case for the diamond-modality is similar.

$\square$

References

[Andersen, 1993] Andersen, H. R. (1993). *Verification of Temporal Properties of Concurrent Systems.* PhD thesis, Department of Computer Science, Aarhus University, Denmark. PB-445.

[Andersen, 1994] Andersen, H. R. (1994). Model checking and Boolean graphs. *Theoretical Computer Science*, 126(1):3–30.

[Andersen, 1995] Andersen, H. R. (1995). Partial model checking (extended abstract). In *Proceedings, Tenth Annual IEEE Symposium on Logic in Computer Science*, pages 398–407, La Jolla, San Diego. IEEE Computer Society Press.

[Andersen and Winskel, 1992] Andersen, H. R. and Winskel, G. (1992). Compositional checking of satisfaction. *Formal Methods in System Design*, 1(4).

[Arnold and Crubille, 1988] Arnold, A. and Crubille, P. (1988). A linear algorithm to solve fixed-point equations on transition systems. *Information Processing Letters*, 29:57–66.

[Bekič, 1984] Bekič, H. (1984). Definable operations in general algebras, and the theory of automata and flow charts. In C.B.Jones, editor, *Hans Bekič: Programming Languages and Their Definition*, volume 177, pages 30–55. Springer-Verlag.

[Cleaveland, 1990] Cleaveland, R. (1990). Tableau-based model checking in the propositional mu-calculus. *Acta Informatica*, 27:725–747.

[Cleaveland et al., 1992] Cleaveland, R., Dreimüller, M., and Steffen, B. (1992). Faster model checking for the modal mu-calculus. In [v. Bochmann and Probst, 1992], pages 383–394.

[Cleaveland and Steffen, 1992] Cleaveland, R. and Steffen, B. (1992). A linear-time model-checking algorithm for the alternation-free modal mu-calculus. In Larsen, K. G. and Skou, A., editors, *Proceedings of the 3rd Workshop on Computer Aided Verification, July 1991, Aalborg*, volume 575 of *LNCS*. Springer-Verlag.

[Emerson and Lei, 1986] Emerson, E. A. and Lei, C.-L. (1986). Efficient model checking in fragments of the propositional mu-calculus. In *Proceedings of Symposium on Logic in Computer Science*, pages 267–278, Cambridge. IEEE.

[Kozen, 1983] Kozen, D. (1983). Results on the propositional mu-calculus. *Theoretical Computer Science*, 27.

[Larsen, 1988] Larsen, K. G. (1988). Proof systems for Hennessy-Milner logic with recursion. In Dauchet, M. and Nivat, M., editors, *Proceedings of CAAP, Nancy, Franch*, volume 299 of *Lecture Notes in Computer Science*, pages 215–230.

[Larsen, 1992] Larsen, K. G. (1992). Efficient local correctness checking. In [v. Bochmann and Probst, 1992].

[Larsen and Xinxin, 1990] Larsen, K. G. and Xinxin, L. (1990). Compositionality through an operational semantics of contexts. In Paterson, M., editor, *Proceedings of ICALP*, volume 443 of *LNCS*, pages 526–539. Springer-Verlag.

[Milner, 1989] Milner, R. (1989). *Communication and Concurrency*. Prentice Hall.

[Stirling, 1985a] Stirling, C. (1985a). A complete compositional modal proof system for a subset of CCS. volume 194 of *Lecture Notes in Computer Science*, pages 475–486. Springer-Verlag.

[Stirling, 1985b] Stirling, C. (1985b). A complete modal proof system for a subset of SCCS. volume 185 of *Lecture Notes in Computer Science*, pages 253–266. Springer-Verlag.

[Stirling, 1987] Stirling, C. (1987). Modal logics for communicating systems. *Theoretical Computer Science*, 49:311–347.

[Stirling, 1992] Stirling, C. (1992). Modal and Temporal Logics. In Abramsky, S., Gabbay, D., and Maibaum, T., editors, *Handbook of Logic in Computer Science*, volume 2, pages 477–563. Oxford University Press.

[Stirling and Walker, 1991] Stirling, C. and Walker, D. (1991). Local model checking in the modal mu-calculus. *Theoretical Computer Science*, 89(1):161–177.

[Tarski, 1955] Tarski, A. (1955). A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5:285–309.

[v. Bochmann and Probst, 1992] v. Bochmann, G. and Probst, D. K., editors (1992). *Proceedings of the 4th Workshop on Computer Aided Verification, CAV'92, June 29 - July 1, 1992, Montreal, Quebec, Canada*, volume 663 of *LNCS*. Springer-Verlag.

[Vergauwen and Lewi, 1992] Vergauwen, B. and Lewi, J. (1992). A linear algorithm for solving fixed-point equations on transition systems. In Raoult, J.-C., editor, *Proceedings of 17'th Colloquium on Trees in Algebra and Programming, CAAP'92, Rennes, France*, volume 581 of *LNCS*, pages 322–341. Springer-Verlag.

[Winskel, 1984] Winskel, G. (1984). Synchronisation trees. *Theoretical Computer Science*, 34:33.

[Winskel, 1985] Winskel, G. (1985). On the composition and decomposition of assertions. Technical Report TR-59, Computer Laboratory, University of Cambridge.

[Winskel, 1986] Winskel, G. (1986). A complete proof system for SCCS with modal assertions. *Fundamenta Informaticae*, IX:401–420.

[Winskel, 1989] Winskel, G. (1989). A note on model checking the modal ν-calculus. In Ausiello, G., Dezani-Ciancaglini, M., and Rocca, S. R. D., editors, *Proceedings of ICALP*, volume 372 of *LNCS*, pages 761–772. Springer-Verlag.

[Winskel, 1990a] Winskel, G. (1990a). A compositional proof system on a category of labelled transition systems. *Information and Computation*, 87.

[Winskel, 1990b] Winskel, G. (1990b). On the compositional checking of validity. In Baeten, J. and Klop, J., editors, *Proceedings of CONCUR '90*, volume 458 of *LNCS*, pages 481–501. Springer-Verlag.

Recent BRICS Report Series Publications

RS-98-40 Henrik Reif Andersen, Colin Stirling, and Glynn Winskel. *A Compositional Proof System for the Modal μ-Calculus*. December 1998. 29 pp.

RS-98-39 Daniel Fridlender. *An Interpretation of the Fan Theorem in Type Theory*. December 1998. 15 pp. To appear in *International Workshop on Types for Proofs and Programs 1998*, TYPES '98 Selected Papers, LNCS, 1999.

RS-98-38 Daniel Fridlender and Mia Indrika. *An n-ary `zipWith` in Haskell*. December 1998. 12 pp.

RS-98-37 Ivan B. Damgård, Joe Kilian, and Louis Salvail. *On the (Im)possibility of Basing Oblivious Transfer and Bit Commitment on Weakened Security Assumptions*. December 1998. 22 pp. To appear in *Advances in Cryptology: International Conference on the Theory and Application of Cryptographic Techniques*, EUROCRYPT '99 Proceedings, LNCS, 1999.

RS-98-36 Ronald Cramer, Ivan B. Damgård, Stefan Dziembowski, Martin Hirt, and Tal Rabin. *Efficient Multiparty Computations with Dishonest Minority*. December 1998. 19 pp. To appear in *Advances in Cryptology: International Conference on the Theory and Application of Cryptographic Techniques*, EUROCRYPT '99 Proceedings, LNCS, 1999.

RS-98-35 Olivier Danvy and Zhe Yang. *An Operational Investigation of the CPS Hierarchy*. December 1998.

RS-98-34 Peter G. Binderup, Gudmund Skovbjerg Frandsen, Peter Bro Miltersen, and Sven Skyum. *The Complexity of Identifying Large Equivalence Classes*. December 1998. 15 pp.

RS-98-33 Hans Hüttel, Josva Kleist, Uwe Nestmann, and Massimo Merro. *Migration = Cloning ; Aliasing (Preliminary Version)*. December 1998. 40 pp. To appear in *6th International Workshop on the Foundations of Object-Oriented*, FOOL6 Informal Proceedings, 1998.

RS-98-32 Jan Camenisch and Ivan B. Damgård. *Verifiable Encryption and Applications to Group Signatures and Signature Sharing*. December 1998. 18 pp.