Compositional Verification of CCS Processes

Mads Dam[1] and Dilian Gurov[2]

[1] Dept. of Teleinformatics, Royal Institute of Technology (KTH/IT),
Electrum 204, SE-164 40 Kista, Sweden,
mfd@sics.se
[2] Swedish Institute of Computer Science, Box 1263,
SE-164 29 Kista, Sweden,
dilian@sics.se

Abstract. We present a proof system for verifying CCS processes in the modal μ-calculus. Its novelty lies in the generality of the proof judgements allowing parametric and compositional reasoning in this complex setting. This is achieved, in part, by the use of explicit fixed point ordinal approximations, and in part by a complete separation, following an approach by Simpson, of rules concerning the logic from the rules encoding the operational semantics of the process language.

1 Introduction

In a number of recent papers [1,2,3,4,9] proof-theoretical frameworks for compositional verification have been put forward based on Gentzen-style sequents of the shape $\Gamma \vdash \Delta$, where the components of Γ and Δ are correctness assertions $P : \phi$. Several programming or modelling languages have been considered, including CCS [3], the π-calculus [2], CHOCS [1], general GSOS-definable languages [9], and even a significant core fragment of a real programming language, Erlang [4]. An important precursor to the above papers is [10] which used ternary sequents to build compositional proof systems for CCS and SCCS vs. Hennessy-Milner logic [6].

A key idea is that the use of a general sequent format allows correctness properties $P : \phi$ to be stated and proved in a *parametric* fashion. That is, correctness statements ϕ of a composite program $P(Q_1, Q_2)$, say, can be relativized to correctness statements of the components, Q_1, Q_2. A general rule of subterm cut

$$\frac{\Gamma \vdash Q : \psi, \Delta \quad \Gamma, x : \psi \vdash P : \phi, \Delta}{\Gamma \vdash P[Q/x] : \phi, \Delta} \tag{1}$$

allows such subterm assumptions to be introduced and used for compositional verification.

It is, however, difficult to support temporal properties within such a framework. As is well known [12], logics like LTL, CTL, or CTL* are poorly equipped for compositional reasoning without resort to devices like history or prophecy variables. For this reason, our investigations have tended to focus on logics based, in some form, on the modal μ-calculus in which the recursive properties needed

D. Bjørner, M. Broy, A. Zamulin (Eds.): PSI'99, LNCS 1755, pp. 247–256, 2000.

for property decomposition can more adequately be expressed. In [3] the first author showed one way of realizing a proof system using the subterm cut rule, and built, for the first time, a compositional proof system capable of handling general CCS terms, including those that create new processes dynamically. In [4] we used a similar, though considerably improved, approach to address Erlang.

The approach of [3] suffered from two main shortcomings, however:

1. Though systematic, the embedding of the CCS operational semantics into the proof system was indirect, and allowed only rather weak completeness results to be obtained.
2. The handling of recursive formulas was very syntactic and hedged by complicated side conditions, hiding the essence of our proof-theoretical approach from view.

In this paper both these issues are addressed. First, following an idea by Simpson [9] we fully separate the embedding of the transitional semantics for P from the general handling of the logic by employing process variables and transition assertions of the shape $P \xrightarrow{\alpha} Q$. These assertions provide a semantically explicit bridge between the transitions of P and the one-step modalities of the logic. A similar approach is used to handle the second complication. The essential difficulty is that, to be sound, rates of progress for fixed point formulas appearing in different places in a sequent must be related. To achieve this in a simple and semantically explicit way we employ fixed point approximations using ordinal variables, and ordinal constraints of the shape $\kappa_1 < \kappa_2$.

In the paper we first introduce the modal μ-calculus with explicit ordinal approximations, and we introduce the basic form of judgment used in the proof system. In the absence of process structure such as CCS, models are just standard Kripke models. Correspondingly, the proof system in this case can be seen to provide an account of Gentzen-style logical entailment. The novelty, in this case, lies in the use of ordinal approximations. This fragment of the proof system is introduced in Sect. 3. The key ingredient to release the power of this proof system is a rule of discharge, or termination, which recognizes proofs by well-founded induction. In another paper [5] we introduce a game which embodies such a rule, and show completeness of the resulting proof system by reduction to Kozen's well-known axiomatization [7]. A practical rule of discharge, however, must be local which the game condition of [5] is not. Here, instead, we introduce a local version of the discharge rule which is, we believe, a simple and intuitive approximation of the complete global condition. This local discharge rule is introduced (summarily, in this abstract) in Section 4. A full instantiation of our approach to CCS requires in addition an embedding of the CCS operational semantics into the present Gentzen-style format (following Simpson [9]) plus the subterm cut rule (1). This extension is shown in Section 5, and then in Section 6 we give a rough sketch of a correctness proof of a simple infinite state CCS process.

2 Logic

Formulas ϕ are generated by the following grammar, where κ ranges over a set of *ordinal variables*, α over a set of *actions*, and X over a set of *propositional variables*.

$$\phi ::= \phi \vee \phi \mid \neg\phi \mid \langle\alpha\rangle\phi \mid X \mid \mu X.\phi \mid (\mu X.\phi)^{\kappa}$$

An occurrence of a subformula ψ in ϕ is *positive*, if ψ appears in the scope of an even number of negation symbols. Otherwise the occurrence is negative. The formation of least fixed point formulas of one of the shapes $\mu X.\phi$ or $(\mu X.\phi)^{\kappa}$ is subject to the usual formal monotonicity condition that occurrences of X in ϕ are positive. We use the symbols U and V to range over (unindexed) fixed point formulas $\mu X.\phi$. A formula ϕ is *propositionally closed* if ϕ does not have free ocurrences of propositional variables. Standard abbreviations apply:

$$\begin{aligned} \mathit{false} &= \mu X.X, \\ \mathit{true} &= \neg\mathit{false}, \\ \phi \wedge \psi &= \neg(\neg\phi \vee \neg\psi), \\ [\alpha]\phi &= \neg\langle\alpha\rangle\neg\phi, \\ \nu X.\phi &= \neg\mu X.\neg(\phi[\neg X/X]) \end{aligned}$$

We assume the standard modal μ-calculus semantics [7]:

$$\begin{aligned} \|\phi \vee \psi\|\rho &= \|\phi\|\rho \cup \|\psi\|\rho \\ \|\neg\phi\|\rho &= \mathcal{S}\backslash\|\phi\|\rho \\ \|\langle\alpha\rangle\phi\|\rho &= \{P \mid \exists Q \in \|\phi\|\rho.P \xrightarrow{\alpha} Q\} \\ \|X\|\rho &= \rho(X) \\ \|\mu X.\phi\|\rho &= \bigcap\{S \mid S \subseteq \|\phi\|\rho[S/X]\} \end{aligned}$$

augmented by the clause:

$$\|(\mu X.\phi)^{\kappa}\|\rho = \begin{cases} \emptyset & \text{if } \rho(\kappa) = 0 \\ \|\phi\|\rho[\|(\mu X.\phi)^{\kappa}\|\rho/X, \beta/\kappa] & \text{if } \rho(\kappa) = \beta + 1 \\ \bigcup\{\|(\mu X.\phi)^{\kappa}\|\rho[\beta/\kappa] \mid \beta < \rho(\kappa)\} & \text{if } \rho(\kappa) \text{ is a limit ordinal} \end{cases}$$

where ρ is an interpretation function (environment), mapping ordinal variables to ordinals, and propositional variables to sets of closed process terms, or *states*, from a domain $\mathcal{S}$ ranged over by P.

The use of ordinal approximation hinges on the following results (of which (1) is the well-known Knaster-Tarski fixed point theorem).

Theorem 1.

1. $\|\mu X.\phi\|\rho = \bigcup_{\beta} \|(\mu X.\phi)^{\kappa}\|\rho[\beta/\kappa]$
2. $\|(\mu X.\phi)^{\kappa}\|\rho = \bigcup_{\beta<\rho(\kappa)} \|\phi\|\rho[\|(\mu X.\phi)^{\kappa}\|\rho/X, \beta/\kappa]$

Observe how this casts the properties U and U^κ as existential properties: This is useful to motivate the proof rules for fixed point formulas given below. Observe also that, for countable models, quantification over countable ordinals in Theorem 1 suffices. In the definition below, we extend interpretation functions ρ to map process variables x to closed process terms (states).

Definition 1 (Assertions, Judgements).

1. *An* assertion *is an expression of one of the forms $E : \phi$, $\kappa < \kappa'$, or $E \xrightarrow{\alpha} F$, where E, F are a process terms and ϕ is a propositionally closed formula.*
2. *The assertion $E : \phi$ is* valid for an interpretation function ρ *(written $E \models_\rho \phi$), if $E\rho \in \|\phi\|\rho$. The assertion $\kappa < \kappa'$ is valid for ρ, if $\rho(\kappa) < \rho(\kappa')$. The assertion $E \xrightarrow{\alpha} F$ is valid for ρ, if $E\rho \xrightarrow{\alpha} F\rho$ is a valid transition.*
3. *A* sequent *is an expression of the form $\Gamma \vdash \Delta$, where Γ and Δ are sets of assertions.*
4. *The sequent $\Gamma \vdash \Delta$ is* valid *(written $\Gamma \models \Delta$), if for all interpretation functions ρ, all assertions in Γ are valid for ρ only if some assertion in Δ is valid for ρ as well.*

An assertion of the shape $E : \phi$ is called a *property assertion*, an assertion of the shape $\kappa < \kappa'$ is called an *ordinal constraint*, and an assertion of the shape $E \xrightarrow{\alpha} F$ is called a *transition assertion.*

3 Proof System: Logical Entailment

We first consider the problem of logical entailment. In this case, process terms E in assertions of the shape $E : \phi$ are variables.

Structural Rules. We assume the axiom rule, the rule of cut, and weakening:

$$\textsc{Ax}\ \frac{}{\Gamma, A \vdash A, \Delta}$$

$$\textsc{Cut}\ \frac{\Gamma \vdash A, \Delta \quad \Gamma, A \vdash \Delta}{\Gamma \vdash \Delta}$$

$$\text{W-L}\ \frac{\Gamma \vdash \Delta}{\Gamma, A \vdash \Delta} \quad \text{W-R}\ \frac{\Gamma \vdash \Delta}{\Gamma \vdash A, \Delta}$$

As in [9], in the axiom rule assertion A needs only be instantiated to transition assertions, and then Δ can be assumed to be empty. Since Γ and Δ are sets, structural rules like permutation and contraction are vacuous. We conjecture that both cut and the weakening rules are admissible.

Logical Rules. In the following listing we assume that $U = \mu X.\phi$.

$$\neg\text{-L}\ \frac{\Gamma \vdash E : \phi, \Delta}{\Gamma, E : \neg\phi \vdash \Delta} \qquad \neg\text{-R}\ \frac{\Gamma, E : \phi \vdash \Delta}{\Gamma \vdash E : \neg\phi, \Delta}$$

$$\vee\text{-L}\ \frac{\Gamma, E : \phi \vdash \Delta \quad \Gamma, E : \psi \vdash \Delta}{\Gamma, E : \phi \vee \psi \vdash \Delta} \qquad \vee\text{-R}\ \frac{\Gamma \vdash E : \phi, E : \psi, \Delta}{\Gamma \vdash E : \phi \vee \psi, \Delta}$$

$$\langle\alpha\rangle\text{-L}\ \frac{\Gamma, E \xrightarrow{\alpha} x, x : \phi \vdash \Delta}{\Gamma, E : \langle\alpha\rangle\phi \vdash \Delta}\ \mathit{fresh}(x)$$

$$\langle\alpha\rangle\text{-R}\ \frac{\Gamma \vdash E \xrightarrow{\alpha} E', \Delta \quad \Gamma \vdash E' : \phi, \Delta}{\Gamma \vdash E : \langle\alpha\rangle\phi, \Delta}$$

$$U\text{-L}\ \frac{\Gamma, E : U^\kappa \vdash \Delta}{\Gamma, E : U \vdash \Delta}\ \mathit{fresh}(\kappa) \qquad U\text{-R}\ \frac{\Gamma \vdash E : \phi[U/X], \Delta}{\Gamma \vdash E : U, \Delta}$$

$$U^\kappa\text{-L}\ \frac{\Gamma, \kappa' < \kappa, E : \phi[U^{\kappa'}/X] \vdash \Delta}{\Gamma, E : U^\kappa \vdash \Delta}\ \mathit{fresh}(\kappa')$$

$$U^\kappa\text{-R}\ \frac{\Gamma \vdash \kappa' < \kappa, \Delta \quad \Gamma \vdash E : \phi[U^{\kappa'}/X], \Delta}{\Gamma \vdash E : U^\kappa, \Delta}$$

The side condition $\mathit{fresh}(x)$ ($\mathit{fresh}(\kappa)$) is intended to mean that x (κ) does not appear freely in the conclusion of the rule.

The rules for unindexed and indexed fixed point formulas are directly motivated by Theorem 1. The lack of symmetry between rules U-L and U-R is not accidental; their symmetric counterparts are in fact admissable.

Ordinal Constraints. Finally, we need to provide rules for reasoning about ordinal constraints.

$$\mathit{OrdTr}\ \frac{\Gamma, \kappa' < \kappa \vdash \kappa'' < \kappa', \Delta}{\Gamma, \kappa' < \kappa \vdash \kappa'' < \kappa, \Delta}$$

This rule is sufficient provided ordinal variables and constraints are only being introduced during the proof, i.e., do not appear in the root sequent.

Theorem 2 (Local Soundness). *All rules for logical entailment are individually sound: Each rule's conclusion is valid whenever its premises are valid.*

4 Proof System: Rule of Discharge

Processes and formulas can be recursive, allowing for proof trees to grow unboundedly. Intuitively, one would like to terminate an open branch whenever a "repeating" sequent is reached, i.e. a sequent which is an instance, up to some substitution σ, of one of its ancestors, its "companion", in the proof tree. A proof structure, all leaf nodes of which are either axioms or such repeating

nodes, serves as the basis for well-founded ordinal induction arguments. A *global discharge condition* is a sufficient condition for such an argument to be valid. In case a global discharge condition applies all leaves which are not axioms can be considered induction hypothesis instances in some, possibly deeply nested, proof by well-founded induction.

The use of ordinal variables and constraints allows global discharge conditions to be phrased in a clear and semantically transparent way. The most general view of discharge is presented in game-based terms elsewhere [5]. In essence, global discharge guarantees well-foundedness of proofs: That along every infinite path in the infinitely unfolded proof tree, ordinal constraints grow downwards in an unbounded manner.

Here we present a discharge condition which is, in contrast to the global condition of [5], more local, and easier to understand and apply. Moreover, even though it is in general incomplete, it is, in our experience, adequate in a great many situations. In particular it is powerful enough to handle the example considered below.

First a single piece of terminology: Two repeat nodes are called *related* if they are in the same strongly connected component in the directed graph obtained from the proof structure by identifying the repeat nodes with their companions.

Definition 2 (Rule of Discharge). *A node labelled $\Gamma \vdash \Delta$ can be* discharged *with U^κ and substitution σ against an ancestor node labelled $\Gamma' \vdash \Delta'$ if:*

(i) U^κ *occurs as subformula in* Γ' *or* Δ'*;*
(ii) $\phi\sigma \in \Gamma$ *whenever* $\phi \in \Gamma'$*, and* $\phi\sigma \in \Delta$ *whenever* $\phi \in \Delta'$*;*
(iii) $\Gamma \vdash \kappa\sigma < \kappa$ *is derivable;*
(iv) assuming the related discharge nodes labelled $\Gamma_1 \vdash \Delta_1 \ldots \Gamma_n \vdash \Delta_n$ *have been discharged with* $U_1^{\kappa_1} \ldots U_n^{\kappa_n}$ *and* $\sigma_1 \ldots \sigma_n$ *against* $\Gamma_1' \vdash \Delta_1' \ldots \Gamma_n' \vdash \Delta_n'$*, there is a linear ordering* $\prec$ *on these discharge nodes including the present node, such that whenever* $i \prec j$*: (a)* $U_i^{\kappa_i}$ *occurs as subformula in* Γ_j' *or* Δ_j'*, and (b) either* $\kappa_i\sigma_j = \kappa_i$*, or* $\Gamma_j \vdash \kappa_i\sigma_j < \kappa_i$ *is derivable.*

In clause (*iv*), the linear ordering can be chosen differently each time the rule is applied (and a new node is added to the corresponding class of related discharge nodes). The purpose of the clause is to guarantee that along every infinite path in the infinitely unfolded proof tree, ordinal constraints grow downwards in an unbounded manner.

Theorem 3 (Soundness). *The proof system including the rules for logical entailment and the rule of discharge is sound: All sequents derivable in the proof system are valid.*

For finite state labelled transition systems the above proof system reduces to an ordinary model checker like the one presented in [11], and is hence complete for such systems. In general, however, due to undecidability of the model checking problem addressed here, the system is necessarily incomplete.

5 Proof System: Operational Semantics

Having transition assertions allows the transitional semantics of a process language to be embedded directly into the proof system as a separate set of proof rules. This can be done in a straightforward manner for any GSOS-definable language [9]. Here we illustrate this approach on a well-known process language, Milner's Calculus of Communicating Systems [8].

We assume that CCS process terms E are generated by the following grammar, where l ranges over a given set of *labels*, L over subsets of this set of labels, α over *actions* of the shape τ, l or $\overline{l}$, and x over a set of *process variables*.

$$E ::= 0 \mid \alpha.E \mid E + E \mid E|E \mid E\backslash L \mid x \mid \mathit{fix}\, x.E$$

The set of states $\mathcal{S}$ used in Section 2 is the set of all closed process terms. The operational semantics of CCS is given as a closure relation on processes through a set of transition rules [8]: the transitions that a CCS process can perform are exactly those derivable by these rules. Hence, the transition rules can be included directly as right introduction rules into our proof system, while the left introduction rules (stating what transitions are *not* possible), come from the closure assumption.

We present only the most significant of the resulting rules, and in particular the ones used in the Example to follow.

$$\text{0-L}\ \frac{\cdot}{\Gamma, 0 \xrightarrow{\alpha} x \vdash \Delta} \qquad \alpha\text{-R}\ \frac{\cdot}{\Gamma \vdash \alpha.E \xrightarrow{\alpha} E, \Delta}$$

$$\alpha\text{-L-1}\ \frac{\Gamma[E/x] \vdash \Delta[E/x]}{\Gamma, \alpha.E \xrightarrow{\alpha} x \vdash \Delta} \qquad \alpha\text{-L-2}\ \frac{\cdot}{\Gamma, \alpha.E \xrightarrow{\beta} x \vdash \Delta}\ \alpha \neq \beta$$

$$+\text{-L}\ \frac{\Gamma[y/x], E \xrightarrow{\alpha} y \vdash \Delta[y/x] \quad \Gamma[z/x], F \xrightarrow{\alpha} z \vdash \Delta[z/x]}{\Gamma, E + F \xrightarrow{\alpha} x \vdash \Delta}$$

$$+\text{-R}\ \frac{\Gamma \vdash E \xrightarrow{\alpha} E', \Delta}{\Gamma \vdash E + F \xrightarrow{\alpha} E', \Delta}$$

$$|\text{-R-1}\ \frac{\Gamma \vdash E \xrightarrow{\alpha} E', \Delta}{\Gamma \vdash E|F \xrightarrow{\alpha} E'|F, \Delta} \qquad |\text{-R-2}\ \frac{\Gamma \vdash E \xrightarrow{l} E' \quad \Gamma \vdash F \xrightarrow{\overline{l}} F', \Delta}{\Gamma \vdash E|F \xrightarrow{\tau} E'|F', \Delta}$$

$$|\text{-L-1}\ \frac{\Gamma[y|F/x], E \xrightarrow{l} y \vdash \Delta[y|F/x] \quad \Gamma[E|z/x], F \xrightarrow{l} z \vdash \Delta[E|z/x]}{\Gamma, E|F \xrightarrow{l} x \vdash \Delta}$$

$$|\text{-L-2}\ \frac{\begin{array}{c}\Gamma[y_1|F/x], E \xrightarrow{\tau} y_1 \vdash \Delta[y_1|F/x] \\ \Gamma[E|y_2/x], F \xrightarrow{\tau} y_2 \vdash \Delta[E|y_2/x] \\ \Gamma[z_1|z_2/x], l_1 = \overline{l_2}, E \xrightarrow{l_1} z_1, F \xrightarrow{l_2} z_2 \vdash \Delta[z_1|z_2/x]\end{array}}{\Gamma, E|F \xrightarrow{\tau} x \vdash \Delta}$$

$$\mathit{fix}\text{-L}\ \frac{\Gamma, E[fix\, x.E/x] \xrightarrow{\alpha} y \vdash \Delta}{\Gamma, fix\, x.E \xrightarrow{\alpha} y \vdash \Delta} \qquad \mathit{fix}\text{-R}\ \frac{\Gamma \vdash E[fix\, x.E/x] \xrightarrow{\alpha} E', \Delta}{\Gamma \vdash fix\, x.E \xrightarrow{\alpha} E', \Delta}$$

In addition to these rules, a subterm cut rule is needed to allow for parametric and compositional reasoning:

$$\textsc{SubtermCut}\text{-R}\ \frac{\Gamma \vdash F : \psi, \Delta \quad \Gamma, x : \psi \vdash E : \phi, \Delta}{\Gamma \vdash E[F/x] : \phi, \Delta}\ \mathit{fresh}(x)$$

6 Example

Consider a process

$$\mathit{Counter} = \mathit{fix}\, x.\ \mathit{up}.\ (x \mid \mathit{down}.x)$$

which can alternatingly engage in *up* and *down* actions, generating a new copy of itself after each *up* action. Clearly, in any point in time, regardless how many counters have already been spawned, this system can engage in finite sequences of consecutive *down* actions only. This propery can be formalised as the negation of the following formula:

$$\phi = \mu X.\, \neg\, \mu Y.\, \neg\, (\langle \mathit{up} \rangle X \vee \langle \mathit{down} \rangle \neg\, Y)$$

So, we want to prove validity of the sequent

$$\vdash \mathit{Counter} : \neg\, \phi$$

We perform the proof backwards, from this goal sequent towards the axioms, guided by the shape of the formulas and process terms involved. After eliminating the negation and approximating ϕ one obtains

$$\mathit{Counter} : \phi^{\kappa} \vdash \tag{2}$$

Continuing in the same straightforward manner we soon arrive at the following two sequents:

$$\kappa' < \kappa, \mathit{up}.\, (\mathit{Counter} \mid \mathit{down}.\mathit{Counter}) \xrightarrow{\mathit{down}} x \vdash x : \psi$$

$$\kappa' < \kappa, \mathit{Counter} \mid \mathit{down}.\mathit{Counter} : \phi^{\kappa'} \vdash$$

the first of which is an axiom. The second sequent is similar to sequent (2), with the important difference of a new *down.Counter* component having appeared. This is the point where one would like to perform an inductive argument on the system structure, and this can be done using subterm cut. The most important question is what the property of the component being cut is that yields the overall system property being verified. A convenient case is when it is the same property, i.e., when the property being verified composes nicely. This is the case in our example, partly because there is no communication between the components. So, after two applications of subterm cut we obtain the following three sequents:

$$\kappa' < \kappa, \mathit{Counter} : \phi^{\kappa'} \vdash$$
$$\kappa' < \kappa, \mathit{down}.\mathit{Counter} : \phi^{\kappa'} \vdash x : \phi^{\kappa'}$$
$$\kappa' < \kappa, x \mid y : \phi^{\kappa'} \vdash x : \phi^{\kappa'}, y : \phi^{\kappa'}$$

the first of which can be discharged with ϕ^{κ} and substitution $[\kappa \mapsto \kappa']$ against (2). Notice that this node has no related discharge nodes (so far), so only clauses $(i)-(iii)$ of the Rule of Discharge have to be checked in this case. The second sequent is easily reduced to an axiom and a discharge node. Handling the remaining sequent is only slightly more involved.

7 Conclusion

We presented a proof system for verifying CCS processes in the modal μ-calculus. Its novelty lies in the generality of the proof judgements allowing parametric and compositional reasoning, in the complex setting of this powerful logic. This is achieved, in part, by the use of explicit fixed point ordinal approximations, and in part by a complete separation, following Simpson [9], in the proof system of the rules concerning the logic from the rules encoding the operational semantics of the process language (here CCS). This makes the proof system easily adaptable to other languages with a clean transitional semantics.

References

1. R. Amadio and M. Dam. Reasoning about higher-order processes. In *Proc. CAAP'95,* Lecture Notes in Computer Science, 915:202–217, 1995.
2. R. Amadio and M. Dam. A modal theory of types for the π-calculus. In *Proc. FTRTFT'96,* Lecture Notes in Computer Science, 1135:347–365, 1996.
3. M. Dam. Proving properties of dynamic process networks. *Information and Computation*, 140:95–114, 1998.
4. M. Dam, L.-å. Fredlund, and D. Gurov. Toward parametric verification of open distributed systems. In *Compositionality: the Significant Difference,* H. Langmaack, A. Pnueli and W.-P. de Roever (eds.), Lecture Notes Notes in Computer Science, Springer-Verlag, 1536:150–185, 1998.
5. M. Dam and D. Gurov. μ-calculus with explicit points and approximations. In preparation, 1999.
6. M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *Journal of the ACM*, **32**:137–162, 1985.
7. D. Kozen. Results on the propositional μ-calculus. *Theoretical Computer Science*, **27**:333–354, 1983.
8. R. Milner. *Communication and Concurrency.* Prentice Hall International, 1989.
9. A. Simpson. Compositionality via cut-elimination: Hennessy-Milner logic for an arbitrary GSOS. In *Proceedings, Tenth Annual IEEE Symposium on Logic in Computer Science*, pages 420–430, San Diego, California, 26–29 1995. IEEE Computer Society Press.
10. C. Stirling. Modal logics for communicating systems. *Theoretical Computer Science*, 49:311–347, 1987.

11. C. Stirling and D. Walker. Local model checking in the modal mu-calculus. *Theoretical Computer Science*, 89:161–177, 1991.
12. P. Wolper. Temporal logic can be more expressive. *Information and Control*, **56**:72–99, 1983.