A COMPLETE MODAL PROOF SYSTEM FOR A SUBSET OF SCCS *

Colin Stirling
Dept. of Computer Science
Edinburgh University
Edinburgh, U.K.

Abstract

Logical proof systems for concurrent programs are notoriously complex, often involving arbitrary restrictions. One of the main reasons for this is that unlike other major programming concepts parallelism does not appear to have a logical correlate. Here using a simple semantic strategy we tentatively propose one and offer an example modal proof system for a subset of Milner's SCCS. The proof rules are reminiscent of Gentzen introduction rules except that there are also introduction rules for the operators of the program language.

Introduction

Logical proof systems for concurrent programs are notoriously complex, often involving arbitrary restrictions: a representative sample is [AFR,L,LG,OG,ZBR]. There does not seem to be a clean way of reasoning from the parts of a system to the whole. It is very hard to deduce even a weak property of a concurrent system without knowing a good deal about its subcomponents: unexpected interactions can arise and are all too common in practice.

Suppose p is a program and $p \models A$ means that p satisfies (the property expressed by) the formula A. For a standard binary commutative and associative parallel operator $\|$ there is little hope of finding an interesting binary function * on arbitrary formulas which validates an unrestricted implication of the form:

$$\text{if } p \models A \text{ and } q \models B \text{ then } p \| q \models A * B$$

No general method is available for deriving information about a concurrent program from little bits of information about its parts. Unlike other major programming notions parallelism does not appear to have a logical correlate. Here we tentatively propose one and offer an example proof system. The suggestion is a rationalization of ideas in [BK,BKP], and arises from a simple semantic strategy.

Let $\models_B$ be a semantic relation relativized to a formula B which (partially) describes an 'environment': $p \models_B A$ is stipulated to mean

$$\text{for any program } q, \text{ if } q \models B \text{ then } q \| p \models A$$

From $p \models_B A$ and $q \models B$ we can, therefore, derive information about $p \| q$. An immediate consequence of associativity of $\|$ is:

* This work was supported by the Science and Engineering Research Council of the U.K.

$$\text{if } \; p \models_A B \; \text{ and } \; q \models_B C \; \text{ then } \; p \,||\, q \models_A C$$

For suppose r is any program satisfying A then $(r \,||\, p) \,||\, q \models C$ and so also $r \,||\, (p \,||\, q) \models C$. In principle, we have a method of reasoning about concurrent programs with arbitrary numbers of components. Again, in principle, we have a compositional semantics of $||$: in terms of the pair of semantic relations the meaning of a concurrent system can be built from the meanings of its components.

This semantic ploy suggests we introduce two proof-theoretic consequence relations $\vdash$ and $\vdash_B$ to coincide with the semantic pair. Resulting proof rules (introduction rules) for $||$ are then straightforward:

$$\frac{p \vdash B \qquad q \vdash_B C}{p \,||\, q \vdash C} \qquad\qquad \frac{p \vdash_A B \qquad q \vdash_B C}{p \,||\, q \vdash_A C}$$

Such rules, unlike most logics for parallel programs, will not presuppose (as a proof proceeds) either a fixed or a bounded number of potential concurrent subcomponents. They allow one to treat $||$ as a first class program operator on a par with sequential composition ';'. From a program point of view the formula B in $\models_B$ suggests an environment description. Logically, it suggests an assumption : $p \models_B A$ could be written as $B, p \models A$. The proposed proof rules for $||$ are, therefore, analogous to Gentzen's cut rule [G] in the form:

$$\frac{A, \Gamma \vdash B \qquad\qquad B, \Delta \vdash C}{A, \Gamma, \quad \vdash C}$$

(They are also analogous to the sequential composition rule of Hoare logic.)

Using this strategy we offer a sound and complete modal proof system for a subset of Milner's Synchronous Calculus of Communicating Systems, SCCS [Mi3]. The binary parallel operator is a synchronous parallel, a tight coupling. The modal language used is Hennessy-Milner logic [HM1,HM2] which has the virtue, unlike more standard program logics, that its expressiveness is tied to a logic independent criterion, namely bisimulation equivalence. This together with the theoretical simplicity of SCCS considerably aids the development of the modal proof system: the proof rules are introduction rules not only for the logical operators but also for the combinators of the program language. The parallel introduction rules include the pair offered above. This work extends results presented in [St1].

The paper is in five sections followed by a conclusion. Proofs of the results will be contained in a fuller version. The first three sections are introductory: sections 1 and 2 describe Hennessy-Milner logic and its logic independent criterion of expressibility; section 3 outlines the subset of SCCS we build a proof theory for. In section 4 we examine the kind of proof rules we would like and in section 5 the proof system together with example proofs is given.

1. Transition Systems and Bisimulation Equivalence

A nondeterministic or concurrent program may communicate repeatedly with its environment. A simple input/output function is, therefore, too austere as a model of such a program: two programs determining the same input/output function may behave very differently in the same environment. Transition systems have long been recognized as a richer model [K,P1,Si1]. More recently they have been used extensively as models of concurrent programs within the framework of tense (temporal) logic [EH, HS,MP1,MP2,QS]. Here our interest is transition systems whose transitions are labelled.

Definition 1.1 A transition system is a triple $\langle P, Act, \rightarrow \rangle$ where

i. P is a set (of processes)
ii. Act is a set (of actions)
iii. $\rightarrow$ is a mapping which associates with each $a \in Act$ a relation $\xrightarrow{a} \subseteq P \times P$.

A transition system T is finite branching provided that each relation $\xrightarrow{a}$, $a \in Act$, is image finite: $\xrightarrow{a}$ is image finite if for each $p \in P$ the set $\{q \mid p \xrightarrow{a} q\}$ is finite.

In [HM1,HM2] the authors offer, in effect, a very intuitive understanding of a transition system. The set Act is viewed as a set of atomic experiments. An atomic experiment on a process (program) $p \in P$ is understood as an attempt to communicate with p. Communication may change a process depending on its internal structure. The relations $\xrightarrow{a}$ $a \in Act$, are intended to capture the effect of experimentation : $p \xrightarrow{a} q$ means that p can evolve to q in response to an a experiment, or q is the result of a successful a experiment on p. A computation can be viewed as a successful sequence of experiments (communications). Similar ideas are also contained in [DeH,HBR,Ho,Mo].

Hennessy and Milner propose that two processes (programs) should be equivalent (have the same meaning) when no amount of finite experimentation distinguishes them. A formal criterion is offered which is the same as bisimulation equivalence when T is finite branching.

Definition 1.2 A relation $R \subseteq P \times P$ on T is a bisimulation just in case

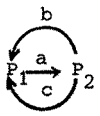
pRq iff i. $\forall a \forall p'$. if $p \xrightarrow{a} p'$ then $\exists q'$. $q \xrightarrow{a} q'$ and $p'Rq'$
ii. $\forall a \forall q'$. if $q \xrightarrow{a} q'$ then $\exists p'$. $p \rightarrow p'$ and $p'Rq'$

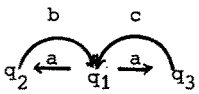
This definition characterizes a property a relation may or may not have on T. (The identity relation, for instance, is a bisimulation.) Such relations give rise to a natural equivalence, bisimulation equivalence, on processes in T:

$$p \sim_T q \text{ iff there exists a bisimulation R such that } pRq$$

It is straightforward to check that $\sim_T$ is an equivalence and, moreover, that it is also the maximal bisimulation under inclusion. Bisimulation equivalence is a very fine equivalence. For instance, consider the transition system given by example 1.3.

Example 1.3





Note that $p_2 \not\sim_T q_2$ and $p_2 \not\sim_T q_3$ because p_2 can respond successfully to both b and c experiments whereas q_2 and q_3 each fail one of the pair. Consequently, $p_1 \not\sim_T q_1$. Thus, if two processes have the same computations (may respond to the same sequences of experiments) this does not guarantee their equivalence. Strong connections between their respective intermediate states are also required. It is precisely these sorts of connections which are, in general, needed for comparing the behaviours of concurrent programs.

Bisimulations have been investigated in [Mi3,Pa,Si2]. Alternative equivalences based on experimental indistinguishability can be found in [Ab,DeH,He2,HBR,Mo,RB]. There is a need to allow infinite experiments when modelling fairness. Bisimulation equivalence is then no longer sufficient. The result is even finer equivalences [He1,Mi2]. However, fairness does not arise in the process language we examine later because its parallel operator is that of tight coupling.

2. Hennessy-Milner Logic

Hennessy and Milner present a modal logic which characterizes bisimulation equivalence on finite branching transition systems [HM1,HM2]. We offer here a negation free version of their logic: the avoidance of negation aids the development of the modal proof theory provided in the sequel. Let $T = (P, Act \rightarrow)$ be a transition system and L_T the modal language:

$$A ::= Tr \mid False \mid A \wedge A \mid A \vee A \mid \langle a \rangle A \mid [a]A \quad \text{where} \quad a \in Act$$

L_T is reminiscent of propositional dynamic logic [Ha]. Here, however, only atomic actions appear within the modal operators. Furthermore, unlike dynamic logic, the satisfaction relation $\models$ is defined between processes and formulas. $\models \subseteq P \times L_T$ is the least relation such that:

$p \models Tr$	for all $p \in P$
$p \not\models False$	for all $p \in P$
$p \models A \wedge B$	iff $p \models A$ and $p \models B$
$p \models A \vee B$	iff $p \models A$ or $p \models B$
$p \models \langle a \rangle A$	iff $\exists q, p \xrightarrow{a} q$ and $q \models A$
$p \models [a]A$	iff $\forall q$. if $p \xrightarrow{a} q$ then $q \models A$

The only atomic formulas are Tr and False: Tr stands for true which every process satisfies whereas False does not hold of any process. $p \models \langle a \rangle A$ means that p can evolve under some successful a experiment to a process satisfying A. Likewise, $p \models [a]A$ means that every process which is the result of a successful a experiment on

p satisfies A. In particular, $p \models [a]\text{False}$ means that p is a-deadlocked: no a experiment on p can be successful. This modal language is, therefore, expressively rich; it can say of processes not only what they can do but also what they can't do. This power of distinguishing is required for the characterization of bisimulation equivalence.

Let $L_T(p) = \{A \mid p \models A\}$ then Hennessy and Milner (in effect) prove the following theorem:

Theorem 2.1 If T is finite branching then $L_T(p) = L_T(q)$ iff $p \sim_T q$

The properties expressible in L_T are tied to the distinguising powers of bisimulation equivalence: the logic L_T cannot differentiate processes which are bisimulation equivalent and vice versa. For instance, the processes p_1, q_1 of example 1.3 are distinguishable by L_T formulas:

$$p_1 \models [a](\langle b\rangle \text{Tr} \wedge \langle c\rangle \text{Tr}) \qquad q_1 \not\models [a](\langle b\rangle \text{Tr} \wedge \langle c\rangle \text{Tr})$$
$$p_1 \not\models \langle a\rangle[b]\text{False} \qquad q_1 \models \langle a\rangle[b]\text{False}$$

Thus, a virtue of L_T, unlike programming logics in general, is that its criterion for expressiveness lies completely outside logic. However, we may wonder how this expressiveness can be translated into expression of particular process properties. If Act in T is finite than $\bigwedge_{a\in Act} [a]\text{False}$ expresses deadlock or termination, and its dual $\bigvee_{a\in Act} \langle a\rangle \text{Tr}$ may be said to express a form of liveness. More generally, the formula or set of formulas expressing a particular property like absence of deadlock will depend upon the particular process under consideration. The assumption that T be finite branching in theorem 2.1 can be discarded if infinite disjunctions and conjunctions are allowed in L_T [HS,Mi1]. Further modal characterizations of equivalences can be found in [BR,GS,HS,Mi1].

The description of Hennessy-Milner logic, L_T, here is semantic. Our aim is to develop a proof system on L_T for a particular process language containing a binary commutative and associative parallel operator. The process languages we choose is a subset of Milner's SCCS.

3. A Subset of SCCS

Milner developed the Synchronous Calculus of Communicating Systems, SCCS, as a tractable model of systems which interact synchronously [Mi3]. It is a transition system $T = (P, Act, \longrightarrow)$ whose set of processes P is built up from as few combinators or operators as possible, each of which is intended to capture a distinctive intuitive concept. This makes SCCS ideal for the sort of proof system we wish to develop. Here we only consider a subset of SCCS, namely SCCS with only finite summation, and without restriction and renaming.

Processes in SCCS evolve relative to some universal discrete time. If $p \xrightarrow{a} p'$ and $q \xrightarrow{b} q'$ then the synchronous parallel of p and q responds to the product of the experiments a and b, $a \times b$, and evolves to the parallel of p' and q'. Product of actions is captured by a structure on Act. Milner assumes that $(Act, \times, 1)$ is an abelian monoid: $\times$ is both commutative and associative with 1 as identity. For simplicity we further assume the left cancellation law:

$$\text{if } a \times b = c \times b \text{ then } a = c$$

The right cancellation law also holds because $\times$ is commutative. We abbreviate $a \times b$ to ab. If $ab = c$ then we let $c \backslash a = b$ (and $c \backslash b = a$): by the cancellation laws if $d \backslash e$ exists it is unique.

The set of processes P of SCCS we consider is given by the closed expressions of the following process language where Z ranges over process variables

$$p ::= Z \mid 0 \mid a.p \mid \text{fix } Z.p \mid p + p \mid p \times p \quad \text{with} \quad a \in Act$$

0 stands for 'disaster': it is a process which cannot respond to any experiment. The process a.p responds to a, and evolves to p. Potentially infinite computations are allowed by the recursion combinator fix Z which binds free occurrences of Z in p in the process fix Z.p. We impose a syntactic restriction on fix Z.p, that Z is guarded in p: that is, every free occurrence of Z in p is within a subexpression a.q of p. Without this restriction the resulting transition system would not be finite branching. The operator + represents external nondeterministic choice: the experimenter may resolve the choice. Finally, $\times$ represents synchronous parallelism.

The remaining undefined feature of the transition system T is the transition relation $\longrightarrow$. It is defined as the least set such that:

$a.p \xrightarrow{a} p$

$\text{fix } Z.p \xrightarrow{a} p'$ whenever $p[\text{fix } Z.p/Z] \xrightarrow{a} p'$ where $[\cdot/\cdot]$ denotes substitution

$p + q \xrightarrow{a} r$ whenever $p \xrightarrow{a} r$ or $q \xrightarrow{a} r$

$p \times q \xrightarrow{bc} p' \times q'$ whenever $p \xrightarrow{b} p'$ and $q \xrightarrow{c} q'$

The process a.p can only respond to an a experiment and in so doing evolves to p. In $a.p + b.q$, where $a \neq b$, the experimenter may resolve the choice: the offer of an a experiment results in p whereas the offer of a b experiment results in q. This is not true of $a.p + a.q$: the experimenter has no control on whether p or q is the result of an a experiment. The number of concurrent subprocesses may increase in response to an experiment. This only happens when the concurrent combinator $\times$ occurs within the scope of a fix Z; for instance, if $p = \text{fix } Z(aZ \times bZ)$ then $p \xrightarrow{ab} p \times p$. This possibility of growth must be reflected in any logical proof system for this language of processes. For a full discussion of SCCS with examples see [Mi3].

Bisimulation equivalence is not only an intuitively natural equivalence on the transition system T it is also a congruence: process contexts preserve equivalence [Mi3]. The following implies that T is finite branching.

Fact 3.1 $\forall p.\ \{q \mid \exists a.\ p \stackrel{a}{\Rightarrow} q\}$ is finite

By theorem 2.1 we know that the modal logic L_T characterizes bisimulation equivalence, $\sim_T$, on T. The following fact states that the parallel operator $\times$ is both commutative and associative up to $\sim_T$ [Mi3].

Fact 3.2 i. $p \times q \sim_T q \times p$

ii. $p \times (q \times r) \sim_T (p \times q) \times r$

This means that formulas of L_T cannot distinguish between these equivalent processes.

The process $\mathbb{0}$ cannot respond to any experiment. It is, therefore, deadlocked: $\mathbb{0} \models [a]\mathrm{False}$ for every $a \in \mathrm{Act}$. Moreover so is $\mathbb{0} \times p$ for any p. As remarked in section 1 processes which may respond to the same sequences of experiments need not be bisimulation equivalent. The following example illustrates this where $\bar{a}$ is the inverse of a; that is $a\bar{a} = 1$.

Example 3.3 $p_i \times q \not\sim_T p_j \times q$ for $1 \leq i < j \leq 3$ where

$q = \mathrm{fix}\ Z.\ \bar{a}.\ Z$

$p_1 = \mathrm{fix}\ Z.\ a.\ Z$

$p_2 = a(a.\mathbb{0} + \mathrm{fix}\ Z.a.Z)$

$p_3 = \mathrm{fix}\ Z.(a.\mathbb{0} + a.Z)$

The three processes $p_i \times q$, $1 \leq i \leq 3$ satisfy every formula in the set $\{\langle 1\rangle \mathrm{Tr}, \langle 1\rangle\langle 1\rangle \mathrm{Tr}, \ldots\}$. (Note the only experiment they can ever respond to is 1.) However, they are not bisimulation equivalent because of differences in possibilities of deadlock. The process $p_1 \times q$ is deadlock free whereas $p_2 \times q$ can only deadlock in one circumstance unlike $p_3 \times q$ which can always deadlock. The process $p_3 \times q$ satisfies every formula in the set $\{\langle 1\rangle[1]\mathrm{False}, \langle 1\rangle\langle 1\rangle[1]\mathrm{False}, \ldots\}$ whereas $p_2 \times q$ only satisfies $\langle 1\rangle\langle 1\rangle[1]\mathrm{False}$ and fails the rest; $p_1 \times q$, on the other hand, fails them all.

4. Towards a Modal Proof Theory: A Relativized Satisfaction Relation

Our aim is to offer a sound and complete modal proof system on L_T for the subset of SCCS outlined. We want, therefore, to define a proof-theoretic consequence relation $\vdash$ which coincides with $\models$. Ideally, the proof rules will be Gentzen style introduction rules [G]. But, we also need to take account of the structure of processes. The theoretical simplicity of the process language suggests that we also offer introduction rules for the combinators. The question then arises as to what, if anything, is the logical correlate of the combinators. In [St1] we provided proof rules for an even more restricted process language, a language devoid of both concurrency and recursion.

Introduction rules for . in a.p. are straightforward. The following schemas suffice:

$$\frac{p \vdash A}{a.p \vdash \langle a\rangle A} \qquad \frac{p \vdash A}{a.p \vdash [a]A}$$

Their justification is that a.p. evolves to p under any a experiment. The logical correlate of . is, therefore, modal iteration. Consequently, these are also <a> and [a] introduction rules.

A global + introduction rule of the form

$$\frac{p \vdash A \qquad q \vdash B}{p + q \vdash f(A,B)}$$

where f is truth-functional and not the constant true function is always unsound: this is shown in [St1]. Restricted versions of such a rule, however, which depend on the form of A and B can be found:

$$\frac{p \vdash \langle a\rangle A}{p + q \vdash \langle a\rangle A} \qquad \frac{q \vdash \langle a\rangle A}{p + q \vdash \langle a\rangle A} \qquad \frac{p \vdash [a]A \qquad q \vdash [a]A}{p + q \vdash [a]A}$$

Their justification is that p + q only evolves to a process which either p evolves to or q evolves to. A restricted metalogical 'or' or 'and' is the correlate of + : if <a>A is true of p or of q it is true of p + q; if [a]A is true of p and of q it is true of p + q.

A global × introduction rule suffers the same fate as a + global rule. Unlike the + case, however, restricted versions which depend on the forms of A and B are inadequate. Such rules would also need to take into account most, if not all, the modal subformulas of A and B. Even if such rules could be found they would be in opposition to the style of proof rules we are suggesting. An alternative approach, a rationalization of [BK,BKP], which fits in with the style of rules suggested already, is now offered. This approach was outlined in the introduction.

We complicate the semantics of L_T, when T is the subset of SCCS outlined, by introducing a relativized satisfaction relation $\models_A$ where A is a formula of L_T. We stipulate that

$$p \models_A B \quad \text{iff} \quad \forall q.\ \text{if } q \models A \ \text{ then } \ q \times p \models B$$

The pair of semantic relations, $\models$, $\models_A$ gives a compositional semantics for concurrency: the samentics of a concurrent system are built up from the semantics of the components. Recall Fact 3.2 that × is both commutative and associative (up to $\sim_T$). By commutativity if $q \models A$ and $p \models_A B$ then also $p \times q \models B$. By associativity the following holds

Fact 4.1 If $p \models_A B$ and $q \models_B C$ then $p \times q \models_A C$

Consequently, if q is the concurrent process with components p_j $0 \leq j \leq n$, in any order and $p_0 \models A_0$ and $p_i \models_{A_{i-1}} A_i$ $1 \leq i \leq n$ then $q \models A_n$.

This semantic ploy suggests that we introduce a second proof-theoretic consequence relation $\vdash_A$. Natural introduction rules for $\times$ then arise:

$$\frac{q \vdash A \qquad p \vdash_A B}{q \times p \vdash B} \qquad \frac{q \vdash A \qquad p \vdash_A B}{p \times q \vdash B}$$

and

$$\frac{q \vdash_A B \qquad p \vdash_B C}{q \times p \vdash_A C} \qquad \frac{q \vdash_A B \qquad p \vdash_B C}{p \times q \vdash_A C}$$

Computationally, A in $p \models_A B$ can be viewed as an environment description, a (possibly partial) summary of any process q such that $q \times p$ satisfies B. Logically, A can be viewed as an assumption : $p \models_A B$ could be rewritten A, $p \models B$. This suggests a logical correlate of $\times$ introduction - in fact, a logical analogy - namely Gentzen's cut rule [G] in the form

$$\frac{A, \Gamma \vdash B \qquad B, \Delta \vdash C}{A, \Gamma, \Delta \vdash C}$$

There is also a similarity to Hoare's (introduction) rule for sequential composition (when $p \vdash_A B$ is written $A\{p\}B$). From now on $A,p \models B$ ($A,p \vdash B$) is written instead of $p \models_A B$ ($p \vdash_A B$).

The additional semantic relation means that introduction rules for two consequence relations $\vdash$ and $\vdash_B$ need to be offered. These relations will be connected by the first pair of $\times$ introduction rules above. Introduction rules for the . and + combinators in the context of $\vdash_B$ are straightforward and not dissimilar from above:

$$\frac{A, p \vdash B}{\langle b \backslash a \rangle A, a.p \vdash \langle b \rangle B} \text{ if } b \backslash a \text{ exists} \qquad \frac{A, p \vdash B}{[b \backslash a] A, a.p \vdash [b] B} \text{ if } b \backslash a \text{ exists}$$

$$\frac{A, p \vdash \langle a \rangle B}{A, p+q \vdash \langle a \rangle B} \qquad \frac{A, q \vdash \langle a \rangle B}{A, p+q \vdash \langle a \rangle B} \qquad \frac{A, p \vdash [a]B \qquad A, q \vdash [a]B}{A, p+q \vdash [a]B}$$

Left unmentioned are introduction rules for fix. The behaviour of fix Z.p is however, fully determined by repeated 'unfolding' : an unfolding of fix Z.p is p[fix Z.p/Z]. (Recall that fix Z.p $\xrightarrow{a}$ q whenever p[fix Z.p] $\xrightarrow{a}$ q.) Contextual introduction rules for fix Z are offered which appeal to this unfolding. The rules depend upon the modal degree of a formula A, m(A), which is inductively defined as the maximum depth of modal operators in A:

$$\begin{aligned} m(Tr) &= m(False) = 0 \\ m(A \vee B) &= m(A \wedge B) = \max(m(A), m(B)) \\ m(\langle a \rangle A) &= m([a]A) = 1 + m(A) \end{aligned}$$

If $p \models A$ and $m(A) = n$ then A is a property of p's evolution through at most n processes; a property of computations from p of length at most n. If $p = \text{fix } Z.q$ then A is at

most a property of the nth 'unfolding' of p given that Z is guarded in q. Consequently, we can appeal to standard approximation techniques: when p = fix Z.q then p^n, $n \geq 0$, is defined inductively:

$$p^0 = \mathbb{0}$$
$$p^{n+1} = q[p^n/Z]$$

For instance, if p = fix Z. a.Z then $p^0 = \mathbb{0}$ and $p^n = \overbrace{a.a. \ldots a.}^{n \text{ times}}\mathbb{0}$, $n > 0$. Hence, p^n, $n \geq 0$, can respond in the same way as fix Z.q to any sequence of experimenting whose length is less than or equal to n: such experimenting is summed up in L_T by formulas whose modal degree is less than or equal to n. The outcome is the following pair of rules for p = fix Z.q

$$\frac{p^n \vdash A}{p \vdash A}\ m(A) \leq n \qquad \frac{A,\ p^n \vdash B}{A,\ p \vdash B}\ m(B) \leq n$$

This method of dealing with fix was suggested by Gerardo Costa.

5. A Complete Modal Proof System for T

The full proof system on L_T for T, the subset of SCCS, is now given.

Axioms $p \vdash Tr$ $\quad \mathbb{0} \vdash [a]A$ $\quad A,p \vdash Tr$ $\quad False, p \vdash A$ $\quad A,\mathbb{0} \vdash [a]B$

$a.p \vdash [b]A$ if $a \asymp b$ $\quad A, a.p \vdash [b]B$ if $b \asymp a$ doesn't exist

vI $\dfrac{p \vdash A}{p \vdash A \vee B}$ $\quad \dfrac{p \vdash B}{p \vdash A \vee B}$ $\quad \dfrac{A,p \vdash B}{A,p \vdash B \vee C}$ $\quad \dfrac{A,p \vdash C}{A,p \vdash B \vee C}$ $\quad \dfrac{A,p \vdash C \quad B,p \vdash C}{A \vee B,p \vdash C}$

$\wedge$I $\dfrac{p \vdash A \quad p \vdash B}{p \vdash A \wedge B}$ $\quad \dfrac{A,p \vdash C}{A \wedge B,p \vdash C}$ $\quad \dfrac{B,p \vdash C}{A \wedge B,p \vdash C}$ $\quad \dfrac{A,p \vdash B \quad A,p \vdash C}{A,p \vdash B \wedge C}$

<a>I $\dfrac{p \vdash A}{a.p \vdash \langle a\rangle A}$ $\quad \dfrac{A,p \vdash B}{\langle a \frown b\rangle A, b.p \vdash \langle a\rangle B}$ $a \frown b$ exists

[a]I $\dfrac{p \vdash A}{a.p \vdash [a]A}$ $\quad \dfrac{A,p \vdash B}{[a \frown b]A, b.p \vdash [a]B}$ $a \frown b$ exists

+I<> $\dfrac{p \vdash \langle a\rangle A}{p+q \vdash \langle a\rangle A}$ $\quad \dfrac{q \vdash \langle a\rangle A}{p+q \vdash \langle a\rangle A}$ $\quad \dfrac{A,p \vdash \langle a\rangle B}{A,p+q \vdash \langle a\rangle B}$ $\quad \dfrac{A,q \vdash \langle a\rangle B}{A,p+q \vdash \langle a\rangle B}$

+I[] $\dfrac{p \vdash [a]A \quad q \vdash [a]A}{p+q \vdash [a]A}$ $\quad \dfrac{A,p \vdash [a]B \quad A,q \vdash [a]B}{A,p+q \vdash [a]B}$

xI $\dfrac{p \vdash A \quad A,q \vdash B}{p \times q \vdash B}$ $\quad \dfrac{p \vdash A \quad A,q \vdash B}{q \times p \vdash B}$ $\quad \dfrac{A,p \vdash B \quad B,q \vdash C}{A,p \times q \vdash C}$ $\quad \dfrac{A,p \vdash B \quad B,q \vdash C}{A,q \times p \vdash C}$

fix I $\dfrac{p^n \vdash A}{p \vdash A}$ $p = fix\ Z.q$, $n \geq m(A)$ $\quad \dfrac{A,p^n \vdash B}{A,p \vdash B}$ $p = fix\ Z.q$, $n \geq m(B)$

The axioms and rules are reasonably straightforward. The axioms $p \vdash Tr$ and $A,p \vdash Tr$ hold because every process satisfies Tr. In contrast, the axiom $False,p \vdash A$

holds because no process satisfies False. The axioms for $\mathbb{0}$ depend upon its inability to respond to any experiment. (Recall that $p \models [a]\mathrm{False}$ says that p is a-deadlocked.) The final three axioms are justified by noting that a.p can only respond to a. The $\vee$I and $\wedge$I rules are as expected. Note, however, that these include introduction rules for the 'environment' or assumption formula. The presence of the assumption complicates the <a>I and [a]I rules as expected. Most of the rest of the rules were mentioned in the previous section. We illustrate the use of some of the rules with an example proof where, as before, $\bar{d}$ for any $d \in \mathrm{Act}$, is the inverse of d; $d\bar{d} = 1$.

Example 5.1 $\quad a(b.\mathbb{0} + c.\mathbb{0}) \times (\mathrm{fix}\ Z(\bar{a}.1.z) \times 1.\bar{b}.\mathbb{0}) \vdash [1]\langle 1\rangle \mathrm{Tr}$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\mathbb{0} \vdash \mathrm{Tr}}\langle b\rangle I}{b.\mathbb{0} \vdash \langle b\rangle \mathrm{Tr}}+\langle\rangle I}{b\mathbb{0} + c\mathbb{0} \vdash \langle b\rangle \mathrm{Tr}}[a]I}{a(b\mathbb{0} + c\mathbb{0}) \vdash [a]\langle b\rangle \mathrm{Tr}}\times I \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\mathrm{Tr},\ \bar{a}.1.\mathbb{0} \vdash \mathrm{Tr}}\langle b\rangle I}{\langle b\rangle \mathrm{Trm} \qquad 1.\bar{a}.1.\mathbb{0} \vdash \langle b\rangle \mathrm{Tr}}[1]I}{[a]\langle b\rangle \mathrm{Tr},\ \bar{a}.1.\bar{a}.1.\mathbb{0} \vdash [1]\langle b\rangle \mathrm{Tr}}\mathrm{fix}\,I}{[a]\langle b\rangle \mathrm{Tr},\ \mathrm{fix}\ Z.\bar{a}.1.Z \vdash [1]\langle b\rangle \mathrm{Tr}}\times I \qquad \dfrac{\dfrac{\dfrac{}{\mathrm{Tr},\ \mathbb{0} \vdash \mathrm{Tr}}\langle 1\rangle I}{\langle b\rangle \mathrm{Tr},\ \bar{b}.\mathbb{0} \vdash \langle 1\rangle \mathrm{Tr}}[1]I}{[1]\langle b\rangle \mathrm{Tr}, 1.\bar{b}.\mathbb{0} \vdash [1]\langle 1\rangle \mathrm{Tr}}$$

$$a(b.\mathbb{0} + c.\mathbb{0}) \times (\mathrm{fix}\ Z(\bar{a}.1.Z) \times 1.\bar{b}.\mathbb{0}) \vdash [1]\langle 1\rangle \mathrm{Tr}.$$

In this example proof the final formula $[1]\langle 1\rangle \mathrm{Tr}$ is proved of the process $1.\bar{b}.\mathbb{0}$ relative to the assumption $[1]\langle b\rangle \mathrm{Tr}$. The proof, however, could have proceeded instead by proving the final formula of either of the other two subprocesses (relative to assumptions). For instance, it is straightforward to show the following pair.

$$[\bar{a}]\langle \bar{b}\rangle \mathrm{Tr},\quad a(b.\mathbb{0} + c.\mathbb{0}) \vdash [1]\langle 1\rangle \mathrm{Tr}$$
$$\mathrm{fix}\ Z(\bar{a}.1.Z) \times 1.\bar{b}\mathbb{0} \vdash [\bar{a}]\langle \bar{b}\rangle \mathrm{Tr}$$

The conclusion follows by $\times$I. All concurrent subprocesses then are to this extent on an equal footing.

In the next example we offer a proof of a process whose number of subcomponents grows as it responds to experiments.

Example 5.2 $\quad \mathrm{fix}\ Z(a.Z \times \bar{a}.Z) \vdash \langle 1\rangle\langle 1\rangle \mathrm{Tr}$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\mathbb{0} \vdash \mathrm{Tr}}\langle \bar{a}\rangle I}{\bar{a}.\mathbb{0} \vdash \langle \bar{a}\rangle \mathrm{Tr}}\times I \qquad \dfrac{\dfrac{}{\mathrm{Tr},\ \mathbb{0} \vdash \mathrm{Tr}}\langle 1\rangle I}{\langle \bar{a}\rangle \mathrm{Tr},\ a.\mathbb{0} \vdash \langle 1\rangle \mathrm{Tr}}}{(a.\mathbb{0} \times \bar{a}\mathbb{0}) \vdash \langle 1\rangle \mathrm{Tr}}\langle a\rangle I \qquad \dfrac{\dfrac{\dfrac{}{\mathrm{Tr},\ \mathbb{0} \vdash \mathrm{Tr}}\langle a\rangle I}{\langle 1\rangle \mathrm{Tr},\ a.\mathbb{0} \vdash \langle a\rangle \mathrm{Tr}}\times 1 \qquad \dfrac{\dfrac{}{\mathrm{Tr},\ \mathbb{0} \vdash \mathrm{Tr}}\langle 1\rangle I}{\langle a\rangle \mathrm{Tr},\ \bar{a}.\mathbb{0} \vdash \langle 1\rangle \mathrm{Tr}}}{\langle 1\rangle \mathrm{Tr},\ a.\mathbb{0} \times \bar{a}.\mathbb{0} \vdash \langle 1\rangle \mathrm{Tr}}\langle 1\rangle I}{a.(a.\mathbb{0} \times \bar{a}.\mathbb{0}) \vdash \langle a\rangle\langle 1\rangle \mathrm{Tr}}\times I \qquad \dfrac{}{\langle a\rangle\langle 1\rangle \mathrm{Tr},\ \bar{a}(a.\mathbb{0} \times \bar{a}.\mathbb{0}) \vdash \langle 1\rangle\langle 1\rangle \mathrm{Tr}}$$

$$\dfrac{a.(a.\mathbb{0} \times \bar{a}.\mathbb{0}) \times \bar{a}(.a.\mathbb{0} \times \bar{a}.\mathbb{0}) \vdash \langle 1\rangle\langle 1\rangle \mathrm{Tr}}{\mathrm{fix}\ Z(a.Z \times \bar{a}.Z) \vdash \langle 1\rangle\langle 1\rangle \mathrm{Tr}}\mathrm{fix}\,I$$

In the final example we prove, in effect, the possibility of deadlock. The process is $p_2 \times q$ of example 3.3. This process can only ever respond to 1 experiments. Thus, $\langle 1\rangle\langle 1\rangle[1]\mathrm{False}$ says that it can become deadlocked.

Example 5.3 $a(a.\emptyset + fix\ Z.a.z) \times fix\ Z.\bar{a}.Z \vdash \langle 1\rangle\langle 1\rangle[1]False$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\emptyset \vdash Tr}\langle\bar{a}\rangle I}{\bar{a}.\emptyset \vdash \langle\bar{a}\rangle Tr}\langle\bar{a}\rangle I}{\bar{a}.\bar{a}.\emptyset \vdash \langle\bar{a}\rangle\langle\bar{a}\rangle Tr}fix I \quad \dfrac{\dfrac{}{Tr,\ \emptyset \vdash [1]False}\langle 1\rangle I}{\langle\bar{a}\rangle Tr,\ a.\emptyset \vdash \langle 1\rangle[1]False}+\langle\rangle I}{fixZ.\bar{a}.Z \vdash \langle\bar{a}\rangle\langle\bar{a}\rangle Tr}\times I \quad \dfrac{\langle\bar{a}\rangle Tr,\ a.\emptyset + fixZ.aZ \vdash \langle 1\rangle[1]False}{\langle\bar{a}\rangle\langle\bar{a}\rangle Tr,\ a(a.\emptyset + fixZ.a.Z) \vdash \langle 1\rangle\langle 1\rangle[1]False}\langle 1\rangle I}{a(a.\emptyset + fix\ Z.a.Z) \times fix\ Z.\bar{a}.Z \vdash \langle 1\rangle\langle 1\rangle[1]False}$$

Soundness and Completeness

The modal proof system above is sound. This is the content of the following theorem:

Theorem 5.4 i. if $p \vdash B$ then $p \models B$
ii. if $A, p \vdash B$ then $A,p \models B$

Its proof is standard: the axioms are all valid and the introduction rules preserve validity.

Completeness can be understood in two ways:

weak completeness : if $p \models B$ then $p \vdash B$

strong completeness : weak completeness and if $A,p \models B$ then $A,p \vdash B$

The modal system is weak complete. Constructing a strongly complete system is problematic. One difficulty is that if A is SCCS-valid, true of every SCCS process, then $B,p \models A$ for any B. In fact, there is not a simple dichotomy between weak and strong completeness: a weak completeness result depends on a corresponding strong result to justify $p \times q \models A$ implies $p \times q \vdash A$. The strong completeness result we offer is modulo formula equivalence on SCCS in the case of the relativized relation. Let $|A| = \{p : p \models A\}$.

Theorem 5.5 i. if $p \models A$ then $p \vdash A$
ii. if $A,p \models B$ then $\exists C.\ |C|=|A|$ and $C,p \models B$

The most difficult part of the proof is the following expressibility (or parallel decomposition) lemma

Lemma 5.6 i. if $p \times q \models A$ then $\exists C.\ p \models C$ and $C,q \models A$
ii. if $A,p \times q \models B$ then $\exists C,D.\ |A|=|C|$ and $C,p \models D$ and $D,q \models B$.

Conclusion

A complete compositional modal proof system for a subset of Milner's SCCS has been offered. The parallel introduction rules were suggested by a simple semantic strategy. There is a number of directions for further work. The semantic strategy involves an appeal to a notion of environment. More general work on environments for SCCS type languages is contained in [La]. The current work is also closely

related to [Wi]. More generally, the strategy needs to be tested against other concurrent programming languages and other logical languages. A small step in this direction is [St2] where we show that the strategy works for the more intractable asynchronous parallel operator of Milner's CCS.

There is also the issue of extending the current modal proof system to cover full SCCS. Allowing infinitary summation means the modal logic must become infinitary [HS,Mi1]. Renaming can be dealt with straightforwardly. Restriction is the operator which causes most problems. Preliminary results suggest that only a subset of restriction contexts can be dealt with by a simple extension of the logic.

Acknowledgements

I would like to thank Bob Constable, Matthew Hennessy, Robin Milner, Gordon Plotkin and in particular Gerardo Costa for discussions and helpful comments on the topic of this paper. I would also like to thank Dorothy McKie for typing.

References

LNCS abbreviates Lecture Notes in Computer Science, Springer-Verlag.

[Ab] S. Abramsky. 'Experiments, powerdomains and fully abstract models for applicative multiprogramming', LNCS Vol.158, pp.1-13 (1983).

[AFR] K. Apt, N. Francez and W.de Roever. 'A proof system for communicating sequential processes', TOPLAS pp. 359-385 (1980).

[BK] H. Barringer and R. Kuiper. 'Towards the hierarchical, Temporal logic, specification of concurrent systems',presented at STL/SERC Workshop on the Analysis of Concurrent Systems, Cambridge. (1983).

[BKP] H. Barringer, R. Kuiper and A. Pnueli, 'Now you may compose temporal logic specifications', Proceedings STOC (1984).

[BR] S. Brookes and W. Rounds. 'Behavioural equivalence relations induced by programming logics', LNCS Vol.154 pp. 97-108 (1983).

[DeH] R. de Nicola and M. Hennessy. 'Testing equivalences for processes', in LNCS Vol. 154 pp. 548-560 (1983).

[EH] E. Emerson and J. Halpern. 'Sometimes and not never revisited: on branching versus linear time', pp. 127-140 POPL Proceedings (1983).

[G] G. Gentzen. 'Investigations into logic deduction', in 'The Collected Works of Gerhard Gentzen' ed. Szabo, North-Holland (1969).

[GS] S. Graf and J. Sifakis. 'A modal characterization of observational congruence on finite terms of CCS', IMAG Technical Report No. 402 (and to appear in ICALP '84) (1983).

[Ha] D. Harel. 'First-Order Dynamic Logic' LNCS Vol.68 (1979).

[HBR] C. Hoare, S. Brookes and A. Roscoe. 'A theory of communicating sequential processes', Technical Monograph Prg-16, Computing Lab, University of Oxford (1981).

[He1] M. Hennessy. 'Axiomatizing finite delay operators', Acta Informatica 21, pp. 61-88 (1984).

[He2] M. Hennessy. 'Modelling finite delay operators'. Technical Report CSR-153-83 Dept. of Computer Science, Edinburgh (1983).

[HM1] M. Hennessy and R. Milner. 'On observing nondeterminism and concurrency', LNCS Vol.85, pp. 299-309 (1980).

[HM2] M. Hennessy and R. Milner. 'Algebraic laws for nondeterminism and concurrency' Technical Report CSR-133-83 (and to appear in JACM) (1983).

[Ho] C. Hoare. 'A model for communicating sequential processes'. Technical Monograph, Prg-22, Computing Lab University of Oxford (1982).

[HS] M. Hennessy and C. Stirling. 'The power of the future perfect in program logics', LNCS Vol.176 pp.301-311 (1984).

[K] R. Keller. 'A fundamental theorem of asynchronous parallel computation' , in Parallel Processing ed. T. Feng, Springer-Verlag (1975).

[L] L. Lamport. 'The 'Hoare logic' of concurrent programs', Acta Informatica pp. 21-37 (1980).

[La] K. Larsen. 'A context dependent equivalence between processes'. To appear.

[LG] G. Levin and D. Gries. 'A proof technique for communicating sequential processes', Acta Informatica pp. 281-302 (1981).

[Mi1] R. Milner. 'A modal characterisation of observable machine-behaviour', LNCS Vol. 112 pp. 25-34 (1981).

[Mi2] R. Milner. 'A finite delay operator in synchronous CCS', Technical Report CSR-116-82, Dept. of Computer Science, Edinburgh (1982).

[Mi3] R. Milner. 'Calculi for synchrony and asynchrony', Theoretical Computer Science, pp. 267-310 (1983).

[Mo] E. Moore. 'Gedanken-experiments on sequential machines', in 'Automata Studies' ed. C. Shannon and J. McCarthy, Princeton University Press, pp. 129-153 (1956).

[MP1] Z. Manna and A. Pnueli. 'Temporal verification of concurrent programs : the temporal framework for concurrent programs', in 'The Correctness Problem in Computer Science', ed. R. Boyer and J. Moore, Academic Press, pp. 215-273 (198).

[MP2] Z. Manna and A. Pnueli. 'How to cook a temporal proof system for your pet language', POPL Proceedings pp. 141-154 (1983).

[OG] S. Owicki and D. Gries. 'An axiomatic proof technique for parallel programs I' Acta Informatica pp. 319-340 (1976).

[Pa] D. Park. 'Concurrency and automata on infinite sequences', LNCS Vol.104 (1981).

[P] G. Plotkin. 'A structural approach to operational semantics'. Lecture Notes, Aarhus University (1981).

[QS] J. Queille and J. Sifakis. 'Fairness and related properties in transition systems - a temporal logic to deal with fairness', Acta Informatica 19, pp. 195-220 (1983).

[RB] W. Rounds and S. Brookes. 'Possible futures, acceptances, refusals and communicating processes' , in Proc. FOCS pp. 140-149 (1981).

[Si1] J. Sifakis. 'Unified approach for studying the properties of transition systems', Theoretical Computer Science, pp. 227-258 (1982).

[Si2] J. Sifakis. 'Property preserving homomorphisms of transition systems', Technical Report, IMAG (1982).

[St1] C. Stirling. 'A proof theoretic characterization of observational equivalence' in Procs. FCT-TCS Bangalore (1983) (and to appear in TCS).

[St2] C. Stirling. 'A compositional modal proof system for a subset of CCS'. To appear.

[ZBR] J. Zwiers, A. de Bruin and W. de Roever. 'A proof system for partial correctness of dynamic networks of processes', Technical Report RUU-CS-83-15, Dept. of Computer Science, University of Utrecht (1983).

[Wi] G. Winskel. 'Complete proof systems for SCCS with modal assertions'. To appear.