CCS, liveness, and local model checking in the linear time mu-calculus

Colin Stirling and David Walker
Department of Computer Science
University of Edinburgh
Edinburgh EH9 3JZ, U.K.

Abstract

In this paper we show that observation equivalence on CCS processes preserves temporal properties (drawn from a very general temporal logic encompassing standard linear and branching time logics). Moreover, relative to a progress requirement, we show that CCS processes are live. But it is also very important to be able to verify that a process has or lacks a temporal property. In Section 3 we briefly discuss the idea of local model checking, checking that a particular finitary process has a property. Finally in Section 4 we exhibit a correct model checker for a sublogic, the linear time mu-calculus, of the general temporal logic.

1 Fair Transition Systems and CCS

Operational semantics of programs and systems are commonly defined in terms of labelled transition systems, structures of the form $(\mathbf{P}, \{\stackrel{a}{\Longrightarrow} \mid a \in Act\})$ where $\mathbf{P}$ is a set of processes (or states), Act a set of actions, and for $a \in Act$, $\stackrel{a}{\Longrightarrow}$ is a transition relation on $\mathbf{P}$: $p \stackrel{a}{\Longrightarrow} p'$ expresses that process p becomes p' by performing the action a. However, especially in the case of concurrent systems, what is of interest is their ongoing behaviour rather than individual allowable transitions. Central to this is the idea of a computation, or a run, understood as a maximal path through the transition system. A path through $(\mathbf{P}, \{\stackrel{a}{\Longrightarrow} \mid a \in Act\})$ is a finite or infinite sequence of the form

$$p_0 \stackrel{a_0}{\Longrightarrow} p_1 \stackrel{a_1}{\Longrightarrow} \ldots$$

where each $p_i \in \mathbf{P}$. A path is *maximal* if either it is infinite, or its final process is unable to perform any action. For uniformity, it is usual to guarantee that a maximal path has infinite length by assuming that the union of the transition relations is total: for each $p \in \mathbf{P}$ there are an $a \in Act$ and a q such that $p \stackrel{a}{\Longrightarrow} q$. This practice is followed here.

Not every maximal path through a transition system may count as a run. For the notion of computation may be defined relative to fairness or liveness assumptions. Instead, some maximal paths are defined to be admissible while the rest are discounted. There are numerous techniques for delineating admissibility — see [1,2,3,4] for a sample. Here we shall extend transition systems with a Streett acceptance condition which takes into account actions as well as states. Generalizing [5], we call the resulting structures *fair transition systems.*

Definition 1 A *fair transition system* is a triple $\mathcal{T} = (\mathbf{P}, \{\stackrel{a}{\Longrightarrow} \mid a \in Act\}, \Psi)$ where
(a) $(\mathbf{P}, \{\stackrel{a}{\Longrightarrow} \mid a \in Act\})$ is a transition system,
(b) $\forall p \in \mathbf{P}.\, \exists a \in Act.\, \exists q \in \mathbf{P}.\, p \stackrel{a}{\Longrightarrow} q$,
(c) $\Psi \subseteq 2^{Act \times \mathbf{P}} \times 2^{Act \times \mathbf{P}}$.

The acceptance condition Ψ is therefore a binary relation on subsets of $Act \times \mathbf{P}$. A path $\sigma = p_0 \stackrel{a_0}{\Longrightarrow} p_1 \stackrel{a_1}{\Longrightarrow} \ldots$ through $\mathcal{T}$ is *admissible* if for each pair $(X, Y) \in \Psi$,

$$\text{if } \stackrel{\infty}{\exists} j.\, (a_j, p_{j+1}) \in X \text{ then } \stackrel{\infty}{\exists} k.\, (a_k, p_{k+1}) \in Y.$$

Notice that neither $\mathbf{P}$ nor Act need be finite. Each fair transition system $\mathcal{T}$ determines a set of runs $\Sigma_{\mathcal{T}}$, the admissible computations through $\mathcal{T}$. We let $\Sigma_{\mathcal{T}}(p)$ be the set of runs in $\Sigma_{\mathcal{T}}$ whose initial process is p. In the sequel we shall only be interested in fair transition systems $\mathcal{T}$ with the property that $\Sigma_{\mathcal{T}}(p)$ is nonempty for each $p \in \mathbf{P}$.

The operational semantics of Milner's CCS [6,7,8] are given as labelled transition systems. In fact there is more than one transition system associated with CCS depending on whether or not the silent action τ is observable. In both cases the set $\mathbf{P}$ of processes is the same, built from a constant $\mathbf{0}$ (nil), variables X, and closed under various operations including action prefixing $a.$, nondeterminism $+$, parallel $|$, recursion $\mathtt{fix}X.$, and restriction $\backslash a$. The action sets depend on a little structure. Let Λ be a set of atomic actions and let $\overline{\Lambda}$ be a set of co-actions disjoint from Λ and in bijection with it. The bijection (and its inverse) is $\bar{\ }$, so $\overline{a}$ (in $\overline{\Lambda}$) and a (in Λ) are complementary actions. Complementary actions may synchronize, resulting in τ, the silent action. When τ is viewed as observable the action set is $\Lambda \cup \overline{\Lambda} \cup \{\tau\}$, and the transition relations $\stackrel{a}{\longrightarrow}$ are given by a set of rules which include the following:

$a.\, p \stackrel{a}{\longrightarrow} p$,
if $p \stackrel{a}{\longrightarrow} p'$ then $p + q \stackrel{a}{\longrightarrow} p'$ and $q + p \stackrel{a}{\longrightarrow} p'$,
if $p \stackrel{a}{\longrightarrow} p'$ then $p \mid q \stackrel{a}{\longrightarrow} p' \mid q$ and $q \mid p \stackrel{a}{\longrightarrow} q \mid p'$,
if $p \stackrel{a}{\longrightarrow} p'$ and $q \stackrel{\overline{a}}{\longrightarrow} q'$ then $p \mid q \stackrel{\tau}{\longrightarrow} p' \mid q'$,
if $p[X := \mathtt{fix}X.\, p] \stackrel{a}{\longrightarrow} p'$ then $\mathtt{fix}X.\, p \stackrel{a}{\longrightarrow} p'$,
if $p \stackrel{a}{\longrightarrow} p'$ and $a \notin \{c, \overline{c}\}$ then $p\backslash c \stackrel{a}{\longrightarrow} p'\backslash c$.

There is no rule for the inactive process $\mathbf{0}$. Synchronization of complementary actions shows itself in the fourth rule. In the subsequent rule $p[X := \mathtt{fix}X.\, p]$ denotes p with all free occurrences of X substituted with $\mathtt{fix}X.\, p$. The fruit of this endeavour is the CCS transition system $(\mathbf{P}, \{\stackrel{a}{\longrightarrow} \mid a \in \Lambda \cup \overline{\Lambda} \cup \{\tau\}\})$ where $\mathbf{P}$ is the set of closed process expressions. Bisimulation equivalence on this system is called *strong equivalence* [8].

When τ is not observable then a *different* transition system is associated with CCS, the system $(\mathbf{P}, \{\stackrel{a}{\Longrightarrow} \mid a \in Act\})$ where $\mathbf{P}$ is as above and Act is now the set $\Lambda \cup \overline{\Lambda} \cup \{\varepsilon\}$. The transition relations $\stackrel{a}{\Longrightarrow}$ are defined using the relations $\stackrel{a}{\longrightarrow}$ for $a \in \Lambda \cup \overline{\Lambda} \cup \{\tau\}$. First we have

$$\stackrel{\varepsilon}{\Longrightarrow} = (\stackrel{\tau}{\longrightarrow})^*$$

where $(\stackrel{\tau}{\longrightarrow})^*$ is the reflexive and transitive closure of $\stackrel{\tau}{\longrightarrow}$. Secondly, for $a \in \Lambda \cup \overline{\Lambda}$ we have

$$\stackrel{a}{\Longrightarrow} = \stackrel{\varepsilon}{\Longrightarrow} \circ \stackrel{a}{\longrightarrow} \circ \stackrel{\varepsilon}{\Longrightarrow}$$

where $\circ$ is relational composition. So $\stackrel{a}{\Longrightarrow}$ includes the absorption of finite sequences of silent actions before and after a. Bisimulation equivalence on this CCS transition system is denoted by $\approx$ and is known as *observation equivalence* or *weak equivalence* [8].

Consider now runs of CCS processes (where τ is unobservable). The following is a maximal path from $a.p$ (with $a \in \Lambda \cup \overline{\Lambda}$):

$$a.p \stackrel{\varepsilon}{\Longrightarrow} a.p \stackrel{\varepsilon}{\Longrightarrow} \ldots$$

even though this process makes no progress — it is hung, idling indefinitely. Therefore precluding it as a computation is natural. For each process p let

$$Init(p) = \{(a, q) \mid p \stackrel{a}{\Longrightarrow} q \text{ and } a \neq \varepsilon\}.$$

If $(a, q) \in Init(p)$ then as p becomes q by performing a it has then ceased ticking over. Let $\varepsilon(p)$ be the set of derivatives of p under ε which may cease idling:

$$\varepsilon(p) = \{(\varepsilon, q) \mid p \stackrel{\varepsilon}{\Longrightarrow} q \text{ and } Init(q) \neq \emptyset\}.$$

More generally, associated with CCS is a fair transition system $(\mathbf{P}, \{\stackrel{a}{\Longrightarrow} \mid a \in Act\}, \Psi)$ whose acceptance condition is:

$$\Psi = \{(\varepsilon(p), Init(p)) \mid p \in \mathbf{P}\}.$$

Thus Ψ amounts to the *liveness* condition that processes eventually proceed if they can. The only paths precluded are those of the form

$$p_0 \stackrel{a_0}{\Longrightarrow} \ldots \stackrel{a_{i-1}}{\Longrightarrow} p_i \stackrel{\varepsilon}{\Longrightarrow} \ldots \stackrel{\varepsilon}{\Longrightarrow} p_n \stackrel{\varepsilon}{\Longrightarrow} \ldots$$

where for infinitely many j (and hence for each j) $Init(p_j) \neq \emptyset$. Clearly each process gives rise to a run. This fair transition system is a principal object of our enquiry below. (There is refinement of $\approx$ to a preorder giving an explicit treatment of *divergence*, [9]; this allows a distinction between indefinite idling and infinite internal chatter. The appropriate logics which characterize this preorder are intuitionistic — see [10] for intuitionistic Hennessy-Milner logic.)

2 General Temporal Logics

Temporal logics are suitable for reasoning about the ongoing behaviour of systems. Standard propositional temporal (and modal) logics are covered by the following general temporal logic where Q ranges over atomic sentences, a over actions, and Z over propositional variables:

$$A ::= Q \mid Z \mid \neg A \mid A \wedge A \mid O_a A \mid OA \mid \forall A \mid \nu Z.\, A.$$

The operator O_a is a relativized next operator while O is its unrelativized counterpart. $\forall$ is the 'for all paths' operator, an essential ingredient of a branching time logic. Finally, $\nu Z.$ is the maximal fixpoint operator, binding free occurrences of Z in A. One restriction on $\nu Z.\, A$ is that each free occurrence of Z in A lies within the scope of an even number of negations. Derived operators are defined in the familiar way: $A \vee B$ is $\neg(\neg A \wedge \neg B)$; $\exists A$ is $\neg\forall\neg A$; and $\mu Z.\, A$ is $\neg\nu Z.\, \neg A[Z := \neg Z]$ where $A[Z := \neg Z]$ is the result of substituting $\neg Z$ for each free occurrence of Z in A.

We interpret formulae of this logic (with actions drawn from the set Act) on fair transition systems $\mathcal{T} = (\mathbf{P}, \{\stackrel{a}{\Longrightarrow} \mid a \in Act\}, \Psi)$ where for each $p \in \mathbf{P}$, $\Sigma_{\mathcal{T}}(p)$ is nonempty. A model is a pair $(\mathcal{T}, V)$ where $\mathcal{T}$ is such a fair transition system and V is a valuation assigning sets of processes to atomic propositions, and sets of paths in $\Sigma_{\mathcal{T}}$ to variables: $V(Q) \subseteq \mathbf{P}$ and $V(Z) \subseteq \Sigma_{\mathcal{T}}$. We assume the customary updating notation: $V[\Sigma'/Z]$ is the valuation V' which agrees with V except on Z where $V'(Z) = \Sigma'$. A little more notation: if σ is a maximal path $p_0 \stackrel{a_0}{\Longrightarrow} p_1 \stackrel{a_1}{\Longrightarrow} \ldots$ in $\Sigma_{\mathcal{T}}$ then $\sigma(0)$ is the initial process p_0, $\sigma(\lambda 0)$ is the initial action a_0, and for $i > 0$, σ^i is the i^{th} suffix of σ,

the maximal path (also in Σ_T) $p_i \stackrel{a_i}{\Longrightarrow} p_{i+1} \stackrel{a_{i+1}}{\Longrightarrow} \ldots$. For each formula A we inductively define $\|A\|_V^T$, the set of paths in Σ_T which satisfy A in the model $\mathcal{M} = (T, V)$. For ease of notation we dispense with the index T from $\|A\|_V^T$ and Σ_T:

$$\begin{aligned}
\|Q\|_V &= \{\sigma \in \Sigma \mid \sigma(0) \in V(Q)\} \\
\|Z\|_V &= V(Z) \\
\|\neg A\|_V &= \Sigma - \|A\|_V \\
\|A \wedge B\|_V &= \|A\|_V \cap \|B\|_V \\
\|O_a A\|_V &= \{\sigma \in \Sigma \mid \sigma^1 \in \|A\|_V \text{ and } \sigma(\lambda 0) = a\} \\
\|OA\|_V &= \{\sigma \in \Sigma \mid \sigma^1 \in \|A\|_V\} \\
\|\forall A\|_V &= \{\sigma \in \Sigma \mid \forall \sigma' \in \Sigma. \text{ if } \sigma'(0) = \sigma(0) \text{ then } \sigma' \in \|A\|_V\} \\
\|\nu Z.\, A\|_V &= \bigcup\{\Sigma' \subseteq \Sigma \mid \Sigma' \subseteq \|A\|_{V[\Sigma'/Z]}\}.
\end{aligned}$$

The expected clause for the derived operator $\mu Z.$ is:

$$\mu Z.\, A = \bigcap\{\Sigma' \subseteq \Sigma \mid \|A\|_{V[\Sigma'/Z]} \subseteq \Sigma'\}.$$

Temporal formulae have sets of paths as meanings (in a model). But for most purposes, paths are subordinate to processes. So a useful derived notion is the set of temporal properties which hold of a process p in the model $\mathcal{M} = (T, V)$. This we define as the set $|p|_V^T$:

$$|p|_V^T = \{A \mid A \text{ is closed and } \Sigma_T(p) \subseteq \|A\|_V^T\}.$$

Note here the restriction to closed formulae as expressions of temporal properties.

This logic is very general (and its satisfiability problem is just within the decidable boundary). It contains as sublogics standard linear time logics including Wolper's extended temporal logic, and the standard branching time logics CTL, CTL* and extended CTL*. Moreover it also contains modal logics — the mu-calculus, and hence Hennessy-Milner logic and propositional dynamic logic — where the modality $[a]$ is $\forall \neg O_a \neg$. (It is common with some of these sublogics to distinguish syntactically between path and state formulae. We prefer a more general semantic definition: a state formula A has the property that for any model (T, V)

$$\sigma \in \|A\|_V^T \text{ iff } \Sigma_T(\sigma(0)) \subseteq \|A\|_V^T .$$

A path formula fails this criterion. Hence, for instance, any boolean combination of formulae of the forms $\forall A$ and $\exists A$ is a state formula.)

It is also a very appropriate logic for CCS, especially as a fair transition system. For instance, the strong liveness property that a must happen infinitely often is given by the formula

$$\nu Z.\, (\mu Y.\, O_a \mathtt{true} \vee OY) \wedge OZ$$

(where `true` is an atomic proposition), which holds of the process $\mathtt{fix}X.\, (\tau.\, X + a.\, b.\, X)$. Let CCS stand for of its own fair transition system. We say that the valuation V is $\approx$-*preserving* provided that the atomic sentences cannot distinguish processes that are weakly equivalent: that is, if whenever $p \approx q$ then for any atomic proposition Q, $p \in V(Q)$ iff $q \in V(Q)$. Now a central theorem: that weak equivalence preserves temporal properties in the case of $\approx$-preserving valuations.

Theorem 2 If V is $\approx$-preserving and $p \approx q$, then $\|p\|_V^{\mathrm{CCS}} = \|q\|_V^{\mathrm{CCS}}$.

This means that weak equivalence preserves both *liveness* and safety properties. (An immediate corollary, if we wish to work with the congruence $\approx^c$ [6,8] instead of $\approx$, is the result where $\approx^c$ replaces $\approx$ throughout Theorem 2.) The proof of Theorem 2 is a little delicate, and proceeds via a more general result proved in [11] that extended bisimulation equivalence on any fair transition system preserves temporal properties. But extended bisimulation equivalence on the CCS fair transition system coincides with $\approx$. An additional twist is that temporal properties are also preserved by $\approx$ on the different CCS fair transition system whose acceptance condition is the empty set — where all maximal paths are admissible. But then no process has any interesting liveness properties: for instance, the process $a.q$ fails to have the property 'eventually a must happen' because of the admissible idling computation $a.q \stackrel{\varepsilon}{\Longrightarrow} a.q \stackrel{\varepsilon}{\Longrightarrow} \ldots$.

It has been claimed, contrary to the results here, that $\approx$ does not preserve temporal properties. The CCS τ-law

$$a.(p + \tau.q) \approx a.(p + \tau.q) + a.q$$

is cited as showing this: the idea is that the left hand side satisfies eventually $p+\tau.q$ unlike the right hand side because of the summand $a.q$ (assuming of course that q and $p + a.q$ are not equivalent). But the problem with this analysis is that the notion of path is that given by the *different* CCS transition system when τ is observable. (This is not to deny that there are interesting abstractions of strong equivalence such as stuttering equivalences which are finer than weak equivalence.)

3 Local Model Checking

A major concern, given the generality of this temporal logic, is its local *model checking* problem: given a finitary model $\mathcal{M} = (\mathcal{T}, V)$, does A belong to $\|p\|_V^{\mathcal{T}}$? A model is *finitary* if for each process p, the set $\{q \mid p(\cup_{a \in Act} \stackrel{a}{\Longrightarrow})^* q\}$ is finite. (CCS restricted to the set of processes where $|$ is not within the scope of a $\mathtt{fix}X.$ is finitary.) Notice that the description of the model checking problem here is whether a particular process has or lacks a temporal property even though the semantics of the general temporal logic is given in terms of paths (which are of secondary interest for practical purposes). Moreover, unlike standard accounts of model checking we are not interested in *all* the states or processes of the model with a given property.

The logical structure of temporal formulae directs the model checker presented below — each rule is a connective elimination rule or a fixpoint unrolling rule. However in the case of CCS, there is also the further question of to what extent model checking can be guided by the algebraic theory of processes. For there are two structural dimensions, the algebraic structure of processes and the logical structure of formulae. A simple example of utilizing process structure in model checking is sanctioned by the soundness of the following rules when $a \neq \varepsilon$:

$$\frac{\exists O_a A \in \|p\|_V^{\mathcal{T}}}{\exists O_a A \in \|p + q\|_V^{\mathcal{T}}} \qquad \frac{\exists O_a A \in \|q\|_V^{\mathcal{T}}}{\exists O_a A \in \|p + q\|_V^{\mathcal{T}}}$$

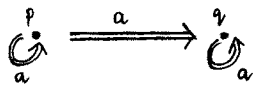
Consequently, if our goal is to discover whether $p + q$ has the property $\exists O_a A$ it suffices to discover if either p or q has it. (For sound and complete sets of such structural rules for CCS in the very restricted case of Hennessy-Milner logic see [10].) Such rules provide a basis for avoiding the 'state explosion problem' [12]. More generally, another technique is to appeal to Theorem 2 (assuming V is $\approx$-preserving) which licences the following rule:

$$\frac{A \in \|p\|_V^{\mathcal{T}} \quad p \approx q}{A \in \|q\|_V^{\mathcal{T}}}$$

Use of this rule is aided by the existence of a minimization algorithm for CCS (which has been automated in the Concurrency Workbench, a toolkit for analysing concurrent systems [13]) that produces a 'smallest' process p from the finitary process q with $p \approx q$. (As it stands, this rule is too coarse — finer analyses may appeal to relativized equivalences $\approx_A$, meaning equivalence up to A, or to equivalences relative to process contexts as in [14].)

4 Local model checking in the linear time mu-calculus

In [15] we present a model checker for a sublogic, the modal mu-calculus, of the general logic of the previous section. An equivalent version of it has been implemented by Cleaveland [16] in the Concurrency Workbench. The crux of the method of model checking is a form of fixpoint induction (inspired by [17]). This is in contrast to Emerson and Lei [18] who use approximation techniques for determining whether or not a process (state) p has a fixpoint property — then one has to determine *all* the processes with that property and then check if p is in that set. Hence, we termed our model checker *local*, because it just tests whether a particular process p has a property A using the structure of A and the processes local to p in the model. Here we provide a local model checker for the linear time mu-calculus, the sublogic without the $\forall$ operator (introduced in [19]; and see also [20]). In later work we hope to include the $\forall$ operator too. Approximation techniques are not applicable. For instance consider the model $\mathcal{M} = (\mathcal{T}, V)$ given by



where $V(Q) = \{q\}$ and $\Psi = \{(\{(a,p)\},\emptyset)\}$. Clearly p has the property 'eventually Q' since the path $(p \stackrel{a}{\Longrightarrow} p)^\omega$ is inadmissible. That is, $\mu Y.\, Q \vee OY \in \|p\|_V^{\mathcal{T}}$. But for all $n \geq 0$, $(\mu Y.\, Q \vee OY)^n \notin \|p\|_V^{\mathcal{T}}$ where for any A, $(\mu Y.\, A)^0 = \mathtt{false}$ and $(\mu Y.\, A)^{n+1} = A[Y := (\mu Y.\, A)^n]$.

4.1 The tableau rules

First we extend the general logic of the previous section to include propositional constants and definition lists. Assume a family of propositional constants ranged over by U. Associated with a constant U is a declaration of the form $U = A$ where A is a closed formula, possibly containing previously declared constant symbols. A *definition list* is a sequence Δ of declarations $U_1 = A_1, \ldots, U_n = A_n$ such that $U_i \neq U_j$ whenever $i \neq j$ and such that each constant occurring in A_i is one of $U_1, \ldots, U_{i-1}$. This means that a prefix of a definition list is itself a definition list. When Δ as above is such a list we let $dom(\Delta) = \{U_1, \ldots, U_n\}$ and $\Delta(U_i) = A_i$. Moreover, if Δ is a definition list, $U \notin dom(\Delta)$ and each constant occurring in A is in $dom(\Delta)$, then $\Delta \cdot U = A$ is the definition list which is the result of appending $U = A$ to Δ. A definition list Δ is *adequate for* B if every constant occurring in B is declared in Δ, and Δ is *adequate for* a set Γ of formulae if it is adequate for each B in Γ. In this circumstance we let B_Δ, resp. Γ_Δ, be the formula B, resp. set of formulae Γ, in the 'environment' Δ (see Definition 3). The interpretation of formulae is now extended to formulae relative to adequate definition lists by, in effect, treating constants as variables.

Definition 3 If $\Delta : U_1 = A_1, \ldots, U_n = A_n$ is adequate for B and Γ, and $V_0 = V$ and $V_{i+1} = V_i[\|A_{i+1}\|_{V_i} / U_{i+1}]$, then

$$\begin{array}{lll} \|B_\Delta\|_V & =_{df} & \|B\|_{V_n} \\ \|\Gamma_\Delta\|_V & =_{df} & \cup\{\|B_\Delta\|_V \mid B \in \Gamma\}. \end{array}$$

The reason for interpreting sets of formulae disjunctively will become clear below. This interpretation accords with the expected meaning of B_Δ and Γ_Δ in terms of syntactic substitution. We define $\Delta^*(B)$, resp. $\Delta^*(\Gamma)$, to be the result of replacing all constants in B, resp. Γ, by their definitions as given in Δ: if Δ is empty then $\Delta^*(B) = B$, and $(\Delta \cdot U_n = A_n)^*(B) = \Delta^*(B[U_n := A_n])$; $\Delta^*(\Gamma) = \cup\{\Delta^*(B) \mid B \in \Gamma\}$.

Lemma 4 $\|B_\Delta\|_V = \|\Delta^*(B)\|_V$ and $\|\Gamma_\Delta\|_V = \|\Delta^*(\Gamma)\|_V$.

Suppose $\mathcal{M} = (\mathcal{T}, V)$ is the finitary model under consideration. The local model checker is a tableau system centred on sequents of the form $p \vdash^{\mathcal{M}}_{\Delta} \Gamma$ where Γ is a finite set of closed linear time mu-calculus formulae, proof-theoretic analogues of

$$\Sigma_{\mathcal{T}}(p) \subseteq \|\Gamma_\Delta\|^{\mathcal{T}}_V .$$

The set Γ is understood disjunctively (see Definition 3). The reason is that although p may have the temporal property $B \vee C$, this does not imply that p has the property B or that p has the property C (for example, B might be 'eventually Q' and C 'eventually R').

As is common in tableau systems, the rules are inverse natural deduction type rules. Each rule is of the form

$$\frac{p \vdash^{\mathcal{M}}_{\Delta} \Gamma}{p_1 \vdash^{\mathcal{M}}_{\Delta_1} \Gamma_1 \quad \cdots \quad p_k \vdash^{\mathcal{M}}_{\Delta_k} \Gamma_k}$$

where $k > 0$, possibly with side-conditions (and side labels). The premise sequent $p \vdash^{\mathcal{M}}_{\Delta} \Gamma$ is the goal to be achieved while the consequents are the subgoals, which are determined by the structure of the model local to p, the definition list Δ, and the structure of the formulae in Γ. Often in the sequel the index $\mathcal{M}$ is dropped from the sequents. In the rules we follow the usual convention of writing Γ, A for $\Gamma \cup \{A\}$.

A tableau for $p \vdash^{\mathcal{M}} A$ is a maximal proof tree whose root is labelled with the sequent $p \vdash^{\mathcal{M}} A$ (where we omit the definition list when, as here, it is empty). The sequents labelling the immediate successors of a node labelled $p \vdash^{\mathcal{M}} A$ are determined by one of the rules. The rules apply only to nodes which are *not* leaves.

Definition 5 A node labelled $p \vdash^{\mathcal{M}}_{\Delta} \Gamma$ is a *leaf* if one of the following holds:

(i) $\Gamma = \emptyset$,
(ii) $\mathtt{true} \in \Gamma$,
(iii) $Q \in \Gamma$ and $p \in V_{\mathcal{M}}(Q)$,
(iv) $\neg Q \in \Gamma$ and $p \notin V_{\mathcal{M}}(Q)$, or
(v) there is a node above it labelled $p \vdash^{\mathcal{M}}_{\Delta'} \Gamma$ (for some Δ').

According to Definition 3, if a leaf fulfills one of the conditions (ii), (iii) or (iv) then it is true, i.e. $\Sigma_{\mathcal{T}}(p) \subseteq \|\Gamma_\Delta\|^{\mathcal{T}}_V$. We call such a leaf *successful*. If it fulfills (i) then it is false and we call it *unsuccessful*. Otherwise it is labelled by a sequent $p \vdash^{\mathcal{M}}_{\Delta} \Gamma$ and above it there is a node labelled by

a very similar sequent $p \vdash^{\mathcal{M}}_{\Delta'} \Gamma$. We call such a leaf a *preterminal* and the associated similar node above it its *companion*. The definition of a successful preterminal is very delicate, and so is delayed until later.

Now for the rules. One technicality is the use of the substitution $B(Z := C)$ which is the result of substituting `true` for all free unguarded occurrences of Z (i.e. those not within the scope of a O or a O_a operator) in B, and substituting C for all free guarded occurrences of Z. For brevity we omit the rules for O_a and $\neg O_a$ which are straightforward. In the O-rule below, $O\Gamma$ is the set of next formulae $\{OA \mid A \in \Gamma\}$. The range of Δ, $rng(\Delta)$, is $\{A \mid \text{ for some } U, \Delta(U) = A\}$.

$$\frac{p \vdash_\Delta \Gamma, Q}{p \vdash_\Delta \Gamma} \qquad \frac{p \vdash_\Delta \Gamma, \neg Q}{p \vdash_\Delta \Gamma}$$

$$\frac{p \vdash_\Delta \Gamma, \neg\mathtt{true}}{p \vdash_\Delta \Gamma} \qquad \frac{p \vdash_\Delta \Gamma, \neg\neg A}{p \vdash_\Delta \Gamma, A}$$

$$\frac{p \vdash_\Delta \Gamma, A \wedge B}{p \vdash_\Delta \Gamma, A \quad p \vdash_\Delta \Gamma, B} \qquad \frac{p \vdash_\Delta \Gamma, \neg(A \wedge B)}{p \vdash_\Delta \Gamma, \neg A, \neg B}$$

$$O\text{-rule} \quad a_1 \frac{}{p_1 \vdash_\Delta \Gamma} \quad p \vdash_\Delta O\Gamma \quad \cdots \quad a_k \frac{}{p_k \vdash_\Delta \Gamma}$$

where $\{(a_1, p_1), \ldots, (a_k, p_k)\} = \{(a, q) \mid p \stackrel{a}{\Longrightarrow} q\}$,

$$\frac{p \vdash_\Delta \Gamma, \neg OA}{p \vdash_\Delta \Gamma, O\neg A}$$

$$\frac{p \vdash_\Delta \Gamma, \nu Z.\, A}{p \vdash_\Delta \Gamma, U} \; \Delta(U) = \nu Z.\, A \qquad \frac{p \vdash_\Delta \Gamma, \neg\nu Z.\, A}{p \vdash_\Delta \Gamma, U} \; \Delta(U) = \neg\nu Z.\, A$$

$$\frac{p \vdash_\Delta \Gamma, \nu Z.\, A}{p \vdash_{\Delta'} \Gamma, U} \; U \notin rng(\Delta) \qquad \frac{p \vdash_\Delta \Gamma, \neg\nu Z.\, A}{p \vdash_{\Delta'} \Gamma, U} \; U \notin rng(\Delta)$$

where $\Delta' = \Delta \cdot U = \nu Z.\, A$ (resp. $\Delta \cdot U = \neg\nu Z.\, A$),

$$\frac{p \vdash_\Delta \Gamma, U}{p \vdash_\Delta \Gamma, A(Z := U)} \; \Delta(U) = \nu Z.\, A \qquad \frac{p \vdash_\Delta \Gamma, U}{p \vdash_\Delta \Gamma, \neg A(Z := \neg U)} \; \Delta(U) = \neg\nu Z.\, A$$

The rules for atomic formulae and booleans are straightforward as is the O-rule. Constants are introduced or reintroduced in the case of fixpoint formulae while the rules for constants unroll the fixpoints they abbreviate. The next (easy) theorem states that any tableau with root labelled $p \vdash^{\mathcal{M}} A$ is finite when $\mathcal{M}$ is finitary.

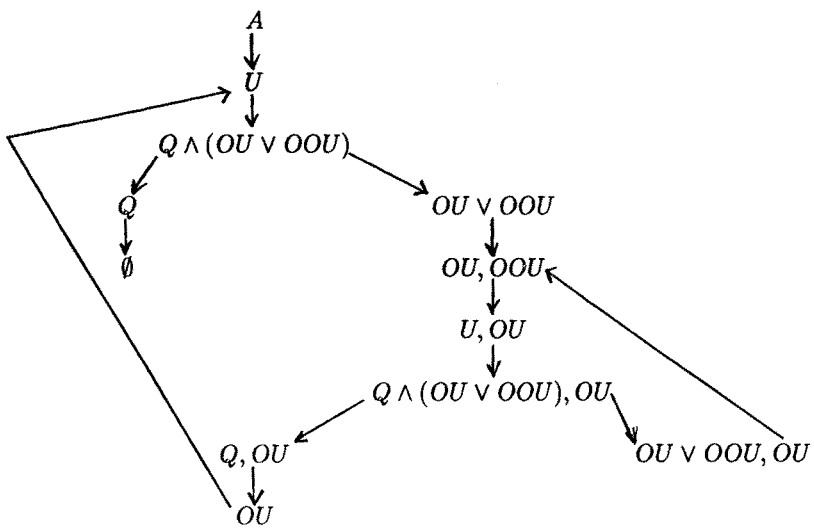
Theorem 6 Every tableau for $p \vdash^{\mathcal{M}} A$ is finite.

4.2 Definition of a successful tableau

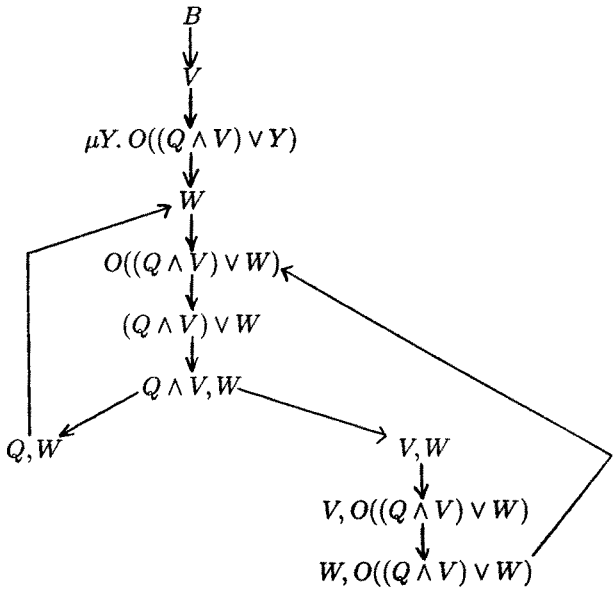
A tableau is *successful* if all its leaves are successful. Above we saw examples of such leaves — for instance a node labelled $p \vdash^{\mathcal{M}}_{\Delta} \Gamma, Q$ with $p \in V_{\mathcal{M}}(Q)$. The outstanding definition is that of a *successful preterminal* which we now present.

With a given formula A we associate a finite directed graph $G(A)$ whose nodes are labelled with finite sets of formulae. This graph has the property that if $\langle p_1 \vdash^{\mathcal{M}}_{\Delta_1} \Gamma_1, \ldots, p_n \vdash^{\mathcal{M}}_{\Delta_n} \Gamma_n \rangle$ is any path through any tableau with root formula A, then $\langle \Gamma_1, \ldots, \Gamma_n \rangle$ is a path through $G(A)$. (So the root of $G(A)$ is labelled $\{A\}$.) The graph $G(A)$ is constructed by examining how sets of formulae evolve under applications of the rules. Rather than describing the construction in full generality we illustrate it with two examples.

1. Let $A \equiv \nu Z.\, Q \wedge (OZ \vee OOZ)$. Then choosing a constant U and setting $\Delta(U) = A$, $G(A)$ (ommiting set braces) is



2. Let $B \equiv \nu Z.\, \mu Y.\, O((Q \wedge Z) \vee Y)$. Then choosing constants V and W and setting $\Delta = \langle V = B, W = \mu Y.\, O((Q \wedge V) \vee Y) \rangle$, $G(B)$ is



Note that since the rules guarantee that no definition list associates two different constants with the same formula we are free to choose the constants to be used in building a tableau when constructing these graphs.

Suppose that there is an edge from Γ to Γ' in such a graph G. As this edge corresponds to the application of one of the rules we can associate with each formula in Γ' at least one formula in Γ of which it is an *immediate descendent*. For example if Γ is $\Gamma_1, A \wedge B$ and Γ' is Γ_1, A, then A in Γ' is an immediate descendent of $A \wedge B$ in Γ, and also of itself if $A \in \Gamma_1$. Every formula in Γ_1 is a descendent of itself. Completing this definition for the remaining rules in the obvious way, given a path $\langle \Gamma_1, \ldots, \Gamma_n \rangle$ through G and formulae A_1 in Γ_1 and A_n in Γ_n, we say that A_n is a *descendent* of A_1 if there are A_i in Γ_i $(i = 2, \ldots n-1)$ such that A_{i+1} is an immediate descendent of A_i for $1 \leq i \leq n-1$.

The next ingredient required for the definition of a successful preterminal is the notion of a constant being active in a formula. Given a formula A, a definition list Δ and a constant U in $dom(\Delta)$, we say that U is *active* in A if either U occurs in A or there is a constant V which occurs in A and is such that U is active in $\Delta(V)$.

Now fix a tableau with root $p \vdash^{\mathcal{M}} A$ say. An *extended path* from a node n to a node n' is a sequence $\pi = \langle n_1, \ldots, n_k \rangle$ of nodes with $n = n_1$ and $n' = n_k$ such that for each i, either n_{i+1} is an immediate successor of n_i in the tableau, or n_i is the immediate predecessor of a preterminal and n_{i+1} is the companion of that preterminal. Associated with such an extended path π is a sequence $p_0 \stackrel{a_0}{\Longrightarrow} p_1 \stackrel{a_1}{\Longrightarrow} \ldots \stackrel{a_{m-1}}{\Longrightarrow} p_m$ of transitions in the model $\mathcal{M}$ arising from the applications of the O-rule on π (note that $p_0 = p_m$). We call this sequence $trans(\pi)$.

Now suppose that $\pi = \langle p_0 \vdash_{\Delta_0} \Gamma_0, \ldots, p_m \vdash_{\Delta_m} \Gamma_m \rangle$ is an extended path from n to n' (so that $p_0 = p_m$ and $\Gamma_0 = \Gamma_m$). The associated path $\tilde{\pi}$ is the path $\langle \Gamma_0, \ldots, \Gamma_m \rangle$ through $G(A)$ obtained by deleting from π everything but the sets of formulae. We say that $\tilde{\pi}$ is *good* if there are $n > 0$ and formulae $B_0, \ldots B_{mn}$ such that $B_0 = B_{mn}$, $B_{jn} \in \Gamma_j$ for $0 \leq j \leq m$, and

1. B_{i+1} is an immediate descendent of B_i for $0 \leq i < m$, and
2. there is U such that $\Delta_0(U) = \nu Z.\, C$ for some Z, C, U is active in each B_i, and for some j, $B_j = U$.

Finally, a preterminal n' is *successful* if for every extended path π from the companion n of n' to n' such that $(trans(\pi))^\omega$ is an admissible path through $\mathcal{M}$, then $\tilde{\pi}$ is good.

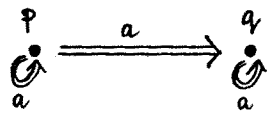
We then have the following result.

Theorem 7 $p \vdash^{\mathcal{M}} A$ has a successful tableau iff $A \in \|p\|^{\mathcal{T}}_V$.

Due to the simplicity of the termination condition the definition of successful preterminal is very complicated. It may be possible to find a simpler definition of successful termination.

4.3 Two examples

Finally we consider two examples. Consider first the earlier example model $\mathcal{M}$

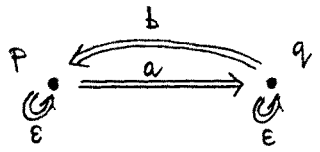


In this example p has the property 'eventually Q'. This is shown by the following successful tableau where $\Delta(U) = \mu Y.\, Q \vee OY$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{p \vdash \mu Y.\, Q \vee OY}{p \vdash_\Delta U}}{p \vdash_\Delta Q \vee OU}}{p \vdash_\Delta Q, OU}}{p \vdash_\Delta OU}}{\begin{array}{ll} a & a \\ p \vdash_\Delta U & \dfrac{\dfrac{q \vdash_\Delta U}{q \vdash_\Delta Q \vee OU}}{q \vdash_\Delta Q, OU} \end{array}}$$

The leaf labelled $p \vdash_\Delta U$ is a preterminal. In this case there is just one inadmissible cycle $(p \stackrel{a}{\Longrightarrow} p)^\omega$. Hence this preterminal is successful.

The second example is the CCS case mentioned earlier. We show that the process $\mathtt{fix}X.\, \tau.\, X + a.\, b.\, X$ in the case of fair CCS has the property 'infinitely often a happens'. This process is p in the following transition system.



In the tableau below $\Delta = \langle V = \nu Z.(\mu Y.\, O_a\mathtt{true} \vee OY) \wedge OZ\rangle$, and $\Delta' = \Delta{\cdot}W = \mu Y.\, O_a\mathtt{true} \vee OY$.

$$\dfrac{\dfrac{\dfrac{\dfrac{p \vdash \nu Z.(\mu Y.\, O_a\mathtt{true} \vee OY) \wedge OZ}{p \vdash_\Delta V}}{p \vdash_\Delta (\mu Y.\, O_a\mathtt{true} \vee OY) \wedge OV}}{\dfrac{p \vdash_\Delta \mu Y.\, O_a\mathtt{true} \vee OY}{p \vdash_{\Delta'} W}}}{\dfrac{p \vdash_{\Delta'} O_a\mathtt{true} \vee OW}{p \vdash_{\Delta'} O_a\mathtt{true}, OW}}$$

$$\varepsilon\,\dfrac{p \vdash_{\Delta'} W}{p \vdash_{\Delta'} W} \qquad a\,\dfrac{q \vdash_{\Delta'} \mathtt{true}, W}{q \vdash_{\Delta'} \mathtt{true}, W}$$

$$\dfrac{\dfrac{q \vdash_\Delta (\mu Y.\, O_a\mathtt{true} \vee OY) \wedge OV}{q \vdash_\Delta \mu Y.\, O_a\mathtt{true} \vee OY}}{q \vdash_{\Delta'} W}$$

$$\dfrac{q \vdash_{\Delta'} O_a\mathtt{true} \vee OW}{q \vdash_{\Delta'} O_a\mathtt{true}, OW}$$

$$\varepsilon\,\dfrac{q \vdash_{\Delta'} W}{q \vdash_{\Delta'} W}$$

$$a\,\dfrac{q \vdash_\Delta V}{q \vdash_\Delta V} \qquad \varepsilon\,\dfrac{p \vdash_\Delta OV}{p \vdash_\Delta V}$$

$$\dfrac{q \vdash_\Delta (\mu Y.\, O_a\mathtt{true} \vee OY) \wedge OV}{q \vdash_\Delta OV}$$

$$\varepsilon\,\dfrac{q \vdash_\Delta OV}{q \vdash_\Delta V} \qquad b\,\dfrac{}{p \vdash_\Delta V}$$

$$b\,\dfrac{}{p \vdash_{\Delta'} W}$$

$$\dfrac{p \vdash_{\Delta'} W}{p \vdash_{\Delta'} O_a\mathtt{true} \vee OW}$$

$$\dfrac{p \vdash_{\Delta'} O_a\mathtt{true} \vee OW}{p \vdash_{\Delta'} O_a\mathtt{true}, OW}$$

$$\varepsilon\,\dfrac{p \vdash_{\Delta'} W}{p \vdash_{\Delta'} W} \qquad a\,\dfrac{q \vdash_{\Delta'} \mathtt{true}, W}{q \vdash_{\Delta'} \mathtt{true}, W}$$

References

[1] D. Lehmann, A. Pnueli and J. Stavi, *Impartiality, justice and fairness: the ethics of concurrent termination*, LNCS 115, 264–277 (1981).

[2] J. Quielle and J. Sifakis, *Fairness and related properties in transition systems*, Acta Informatica 19, 195–220 (1983).

[3] M. Hennessy, *Axiomatising finite delay operators*, Acta Informatica 21, 61-88 (1984).

[4] N. Francez, Fairness, Springer-Verlag (1986).

[5] C. Courcoubetis, M. Vardi and W.Wolper, *Reasoning about fair concurrent programs*, POPL (1986).

[6] R. Milner, A Calculus of Communicating Systems, LNCS 92 (1980).
[7] D. Walker, *Introduction to a Calculus of Communicating Systems*, University of Edinburgh report ECS-LFCS-87-22 (1987).
[8] R. Milner, Communication and Concurrency, Prentice-Hall (1989).
[9] D. Walker, *Bisimulations and Divergence*, Proc. LICS 186–192 (1989).
[10] C. Stirling, *Modal logics for communicating systems*, TCS 49, 311–47 (1987).
[11] M. Hennessy and C. Stirling, *The power of the future perfect in program logics*, Information and Control 67, 23–52 (1985).
[12] E. Clarke and O.Grümberg, *Research on automatic verification of finite-state concurrent systems*, Ann. Rev. Comput. Sci. 2, 269–90 (1987).
[13] R. Cleaveland, J. Parrow and B. Steffen, *The Concurrency Workbench*, this volume.
[14] K. Larsen, *A context-dependent bisimulation between processes* TCS (1988).
[15] C. Stirling and D. Walker, *Local model checking in the modal mu-calculus*, in LNCS 351, 369–383 (1989).
[16] R. Cleaveland, *Tableau based model checking in the propositional mu-calculus*, submitted for publication.
[17] K. Larsen, *Proof systems for Hennessy-Milner logic with recursion*, in Proc. CAAP 88.
[18] A. Emerson and C. Lei, *Efficient model checking in fragments of the propositional mu-calculus*, Proc. LICS, 267–278 (1986).
[19] H. Barringer, R. Kuiper, amd A. Pnueli, *Now you may compose temporal logic specifications*, 16th STOC (1984).
[20] E. Emerson and E. Clarke, *Characterising correctness properties of parallel programs as fixpoints*, LNCS 85 (1981).