BISIMULATIONS AND ABSTRACTION HOMOMORPHISMS

Ilaria Castellani*

Computer Science Department

University of Edinburgh

Abstract

In this paper we show that the notion of bisimulation for a class of labelled transition systems (the class of *nondeterministic processes*) may be restated as one of "reducibility to a same system" via a simple reduction relation. The reduction relation is proven to enjoy some desirable properties, notably a Church-Rosser property. We also show that, when restricted to finite nondeterministic processes, the relation yields unique minimal forms for processes and can be characterised algebraically by a set of reduction rules.

1. Introduction

Labelled transition systems [K,P] are generally recognised as an appropriate model for nondeterministic computations. The motivation for studying such computations stems from the increasing interest in concurrent programming.

When modelling communication between concurrent programs, some basic difficulties have to be faced. A concurrent program is inherently part of a larger environment, with which it interacts *in the course of* its computation. Therefore a simple input-output function is not an adequate model for such a program. The model should retain some information about the internal states of a program, so as to be able to express the program's behaviour in any interacting environment. Also, *nondeterminacy* arises when abstracting from such parameters as the relative speeds of concurrent programs: as a consequence, we need to regard any single concurrent program as being itself nondeterministic.

The question is then to find a model for nondeterministic programs that somehow accounts for intermediate states. On the other hand, only those intermediate states should be considered which are relevant to the "interactive" (or *external*) behaviour of the program. Now one can think of various criteria for selecting such significant states.

In this respect labelled transition systems provide a very flexible model: by varying the definition of the transition relation one obtains a whole range of different descriptions, going from a full account of the structure of a program to some more interesting "abstract" descriptions. However, even these abstract descriptions still need to be factored by equivalence relations (for a review see [B] or [DeN]).

A natural notion of equivalence, *bisimulation equivalence*, has been recently proposed by D. Park [Pa] for transition systems: informally speaking, two systems are said to *bisimulate* each other if a full correspondence can be established between their sets of

---

*Supported by a scholarship from the Consiglio Nazionale delle Ricerche (Italy)

states in such a way that from any two corresponding states the two (sub)systems will still bisimulate each other.

In this paper we show that the notion of bisimulation for a class of labelled transition systems (the class of *nondeterministic processes*) may be restated as one of "reducibility to a same system" via a simple reduction relation. The reduction relation is proven to enjoy some desirable properties, notably a Church-Rosser property. We also show that, when restricted to finite nondeterministic processes, the relation yields unique minimal forms for processes and can be characterised algebraically by a set of reduction rules.

The paper is organised as follows. In section 2 we present our computational model, the class of *nondeterministic processes*. In section 3 we argue that this basic model is not abstract enough, particularly when systems are allowed *unobservable* transitions as well as observable ones. We therefore introduce *abstraction homomorphisms* [CFM] as a means of simplifying the structure of a process by merging together some of its states: the result is a process with a simpler description, but "abstractly equivalent" to the original one. We can then infer a *reduction relation* between processes from the existence of abstraction homomorphisms between them. We prove some significant properties of this relation, such as invariance in contexts and the announced Church-Rosser property. Based on the reduction relation, we define an *abstraction equivalence* relation on processes: two processes are equivalent iff they are both reducible to a same (simpler) process.

In sections 4 and 5 we study the relationship between our notions of reduction and abstraction and the notion of *bisimulation* between transition systems. The criterion we use for identifying states of a process via abstraction homomorphisms is similar to the one underlying the definition of bisimulation: we show in fact that any abstraction homomorphism is a *single-valued bisimulation*. We finally prove that the abstraction equivalence is substitutive in contexts and that it coincides with the *largest* (*substitutive*) *bisimulation*. Our equivalence can then be regarded as a simple alternative formulation for bisimulation equivalence.

In section 6 we consider a small *language* for defining *finite* nondeterministic processes: essentially a subset of R. Milner's CCS (Calculus of Communicating Systems) [M1]. We find that our results combine neatly with some established facts about the language. On this language our equivalence is just Milner's *observational congruence*, for which a complete finite axiomatisation has been given in [HM]. So, on the one hand, we get a ready-made algebraic characterisation for the abstraction equivalence; on the other hand, our characterisation proves helpful in working out a complete system of *reduction rules* for that language. We conclude by proposing a denotational *tree-model* for the language, which is isomorphic to the term-model in [HM].

Most of the results will be stated without proof. For the proofs we refer to the complete version of the paper [C].

2. Nondeterministic Systems

In this section we introduce our basic computational model, the class of *nondeterministic systems*. Nondeterministic systems are essentially labelled transition systems with an initial state.

Let A be a set of elementary *actions* or *transitions*, containing a distinguished symbol τ which denotes a hidden or *unobservable* transition. We will use μ, ν . . . to range over A, and a, b . . . to range over $A - \{\tau\}$.

Definition 2.1: A *nondeterministic system* (*NDS*) over A is a triple $S = (Q\cup\{r\}, A, \longrightarrow)$, where $Q\cup\{r\}$ is the set of *states* of S, $r \notin Q$ is the *initial* state (or root) of S, and $\longrightarrow \subseteq [(Q\cup\{r\}) \times A \times (Q\cup\{r\})]$ is the *transition relation* on S.

We will use q, q' to range over $Q\cup\{r\}$, and write $q \xrightarrow{\mu} q'$ for $(q, \mu, q') \in \longrightarrow$. We interpret $q \xrightarrow{\mu} q'$ as: S may evolve from state q to state q' via a transition μ.

We will also make use of the transitive and reflexive closure $\longrightarrow^*$ of $\longrightarrow$, which we call the *derivation relation* on S. For an NDS $S = (Q\cup\{r\}, A, \longrightarrow)$, we will use Q_S, r_S, $\longrightarrow_S$ instead of Q, r, $\longrightarrow$ whenever an explicit reference to S is required.

According to our definition, an NDS S is a machine starting in some definite state and evolving through successive states by means of elementary transitions. On the other hand, each state of S may be thought of as the initial state of some NDS: then we might regard S as giving rise to new systems, rather than going through successive states.

In fact, if we consider the class $\mathcal{S}$ of all NDS's, we may notice that $\mathcal{S}$ itself can be described as a transition system (although not an NDS, since $\mathcal{S}$ is obviously not rooted). Let $\longrightarrow^*_{\mathcal{S}}$ be the associated derivation relation: we say that S' is a *derivative* of S whenever $S \longrightarrow^*_{\mathcal{S}} S'$. Now it is easy to see that, for any $S \in \mathcal{S}$, a one-to-one correspondence can be established between the states and the derivatives of S. We shall denote by S_q the derivative corresponding to the state q and by $q_{S'}$ the state corresponding to the derivative S'.

In the following we will often avail of this correspondence between states and (sub)systems.

We assume the class $\mathcal{S}$ to be closed w.r.t. some simple *operators*: a nullary operator NIL, a set of unary operators $\mu.$ (one for each $\mu \in A$), and a binary operator +. The intended meaning of these operators is the following: NIL represents *termination*, + is a *free-choice* operator, and the μ's provide a simple form of sequentialisation, called *prefixing* by the action μ.

The transition relation of a compound NDS may be inferred from those of the components by means of the rules:

i) $\mu S \xrightarrow{\mu} S$

ii) $S \xrightarrow{\mu} S'$ implies $S + S'' \xrightarrow{\mu} S'$, $S'' + S \xrightarrow{\mu} S'$

The operators will be given a precise definition for a subclass of $\mathcal{S}$, the class of *nondeterministic processes* that we will introduce in the next section.

2.1 Nondeterministic processes

As they are, NDS's have an isomorphic representation as (rooted) *labelled directed graphs*, whose nodes and arcs represent respectively the states and the transitions of a system. On the other hand, any NDS may be unfolded into an *acyclic* graph. We shall here concentrate on a class of acyclic NDS's that we call *nondeterministic processes* (NDP's).

Basically, NDP's are NDS's whose derivation relation $\longrightarrow^*$ is a *partial ordering*. Each state of a process is assigned a *label*, that represents the sequence of observable actions leading from the root to that state. To make such a labelling consistent, we only allow two paths to join in the graph if they correspond to the same observable derivation sequence. The labelling is subject to the following further restriction: for any label σ, there are at most finitely many states labelled by σ. As it will be made clear subsequently, this amounts to impose a general *image-finiteness* condition on the systems.

In the formal definition, we will use the following notation: A^* is the set of finite sequences over A, with the usual prefix-ordering, and with empty sequence ε . For simplicity the string $<\mu>$ will be denoted by μ. The *covering relation* $-\!\!\subset$ associated to a partial ordering $\leq$ is given by: $x -\!\!\subset y$ iff $x<y$ and $\not\exists$ z such that $x<z<y$. Also, we make the following convention: τ acts as the identity over A^* and will thus be replaced by ε when occurring in strings.

Definition 2.1.1: A *nondeterministic process* (NDP) over A is a triple $P = (Q\cup\{r\}, \leq, l)$ where:

$(Q\cup\{r\}, \leq)$ is a *rooted* poset: $\forall q, \quad r \leq q$

$l: Q\cup \{r\} \longrightarrow A^*$ is a *monotonic* labelling function, satisfying:

$l(r) = \varepsilon$

$q -\!\!\subset q'$ implies $l(q')=l(q).\mu, \ \mu\in A^*$

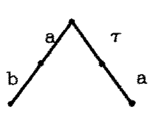
$\forall\ \sigma \in A^*, \quad \{q \mid l(q) = \sigma\}$ is *finite*

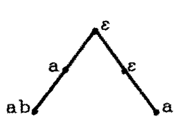
Note that an NDP is very nearly a *labelled tree*: it only differs from a labelled tree in that it might have some confluent paths. The reason we do not directly adopt labelled trees as a model is purely technical (the proof that the model is closed w.r.t. reductions would be rather tricky). However we intend that trees are our real object of interest: in particular, our examples will always be chosen from trees.

As pointed out already, we label nodes with sequences of actions, rather than labelling arcs with single actions: this minor variation w.r.t. the standard notation (see e.g. Milner's *synchronisation trees*) will make it easier to compare different states of a process.

It is easy to see that any NDP P is also an NDS, with $\longrightarrow_P$ given by $-\!\!\subset$. More precisely, for any $\mu\in A$, the relation $\xrightarrow{\mu}_P$ will be given by $\{ (q,q') \mid q -\!\!\subset q' \text{ and } l(q') = l(q).\mu \}$.

Note that, because of our convention that $\tau = \varepsilon$, a τ-transition will be represented in an NDP by the repetition of the same label on the two $\xrightarrow{\tau}$ related nodes. More generally, the label of a node will now represent the sequence of *observable* actions leading to it. For example:

the tree 

will be in our notation: 

In what follows, nondeterministic processes will always be considered up to isomorphism. Formally, an *isomorphism* between two NDP's: $P_1 = (Q_1 \cup \{r_1\}, \leq_1, l_1)$, $P_2 = (Q_2 \cup \{r_2\}, \leq_2, l_2)$ is a one-to-one correspondence: $\Phi : Q_1 \cup \{r_1\} \longrightarrow Q_2 \cup \{r_2\}$ s.t.

i) $l_2(\Phi(q)) = l_1(q)$

ii) $\Phi(q) \leq_2 \Phi(q')$ iff $q \leq_1 q'$

The operators NIL, $\mu.$ and + can be formally defined on NDP's. Let T_i denote the NDP $(Q_i \cup \{r_i\}, \leq_i, l_i)$. Then we have the following :

Definition 2.1.2: (*Operators on NDP's*)

NIL = $(\{r_{NIL}\}, \{(r_{NIL}, r_{NIL})\}, \{(r_{NIL}, \varepsilon)\})$ is the NDP with just a root r_{NIL} and an empty set of subsequent states

μP_1 is the NDP $P = (Q \cup \{r\}, \leq, l)$, where r does not occur in $Q_1 \cup \{r_1\}$, and:

$$Q = Q_1 \cup \{r_1\}$$

$$\leq \; = \; \leq_1 \cup \; \{(r,q) \mid q \in Q\}$$

$$l(q) = \begin{cases} \varepsilon, \text{ if } q = r \\ \mu.l_1(q) \quad \text{otherwise} \end{cases}$$

$P_1 + P_2$ is the NDP $P = (Q \cup \{r\}, \leq, l)$, where r does not occur in $Q_1 \cup Q_2$, and:

$$Q = Q_1 \cup Q_2 \quad \text{(disjoint union)}$$

$$\leq \; = \; \leq_1 \restriction Q_1 \cup \leq_2 \restriction Q_2 \cup \{(r,q) \mid q \in Q\}$$

$$l = l_1 \restriction Q_1 \cup l_2 \restriction Q_2 \cup \{(r, \varepsilon)\}$$

Let $\boldsymbol{P} \subseteq \boldsymbol{S}$ denote the class of all NDP's: in what follows our treatment of nondeterministic systems will be confined to $\boldsymbol{P}$.

3. Abstraction Homomorphisms

The NDP-model, though providing a helpful conceptual simplification, does not appear yet abstract enough. It still allows, e.g., for structural redundancies such as:

Moreover we want to be able, in most cases, to ignore *unobservable* transitions. Such transitions, being internal to a system, should only be detectable indirectly, on account of their capacity of affecting the *observable behaviour* of the system.

We will therefore introduce a *simplification* operation on processes, which we call *abstraction homomorphism*. Essentially an abstraction homomorphism will transform a process in a structurally simpler (but semantically equivalent) process by merging together some of its states.

The criterion for identifying states is that they be *equivalent* in some recursive sense: informally speaking, two states will be equivalent iff they have *equivalent histories* (derivation sequences) and *equivalent futures* or potentials (sets of subsequent states). Formally:

Definition 3.1: Given two NDP's $P_1 = (Q_1 \cup \{r_1\}, \leq_1, l_1)$, $P_2 = (Q_2 \cup \{r_2\}, \leq_2, l_2)$

a function $h: \begin{array}{l} r_1 \longrightarrow r_2 \\ Q_1 \longrightarrow Q_2 \end{array}$ is an *abstraction homomorphism* (a.h.) from P_1 to P_2 iff:

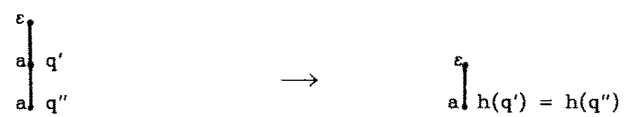
i) $l_2(h(q)) = l_1(q)$

ii) $succ_2(h(q)) = h(succ_1(q))$

where $succ(q) = \{\ q' \mid q \leq q'\ \}$ is the set of successors of q, *inclusive of q*.

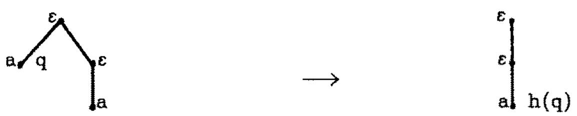
Examples

1)



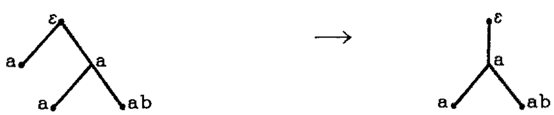
From this example we can see why succ(q) must include q itself: q'' is a *proper* successor of q', whereas h(q'') would not be a proper successor of h(q').

2)

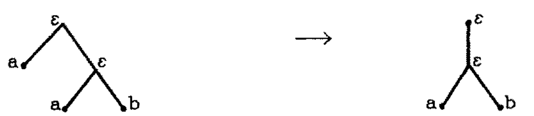


Note that the set of predecessors of q is *not* preserved by the homomorphism.

3)

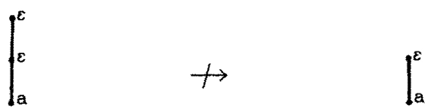


4)

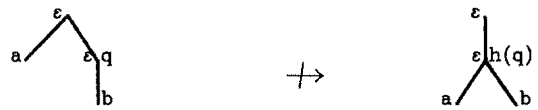


Counterexamples

5)



This example shows that a process of the form τP can only be transformed into a process of the same form. This point will subsequently be made more precise.

6) 

This is not an a.h. because it would increase the set of successors of q.

Abstraction homomorphisms induce the following *reduction relation* $\xrightarrow{abs}$ on processes:

Definition 3.2 : P $\xrightarrow{abs}$ P' iff $\exists$ a.h. h: P $\longrightarrow$ P'.

Since the identity function is an a.h. and the composition of two a.h.'s is again an a.h., the relation $\xrightarrow{abs}$ satisfies the following:

Property 1: $\xrightarrow{abs}$ is reflexive and transitive.

Also, it can easily be shown that:

Property 2: $\xrightarrow{abs}$ is preserved by the operators μ. and +.

We turn now to what is perhaps the most interesting feature of our reduction relation, namely its *confluent* behaviour. Confluence of a.h.'s can be proved by standard algebraic techniques, once the notion of *congruence* associated to an a.h. is formalised.

Definition 3.3: Given an NDP P = $(Q \cup \{r\}, \leq, l)$, we say that an equivalence relation ~ on Q is a *congruence* on P iff, whenever $q \sim q'$:

i) $l(q) = l(q')$ (*labels* are preserved)

ii) $q \leq p$ implies $\exists\ p' \sim p$ s.t. $q' \leq p'$

(*successors* are preserved)

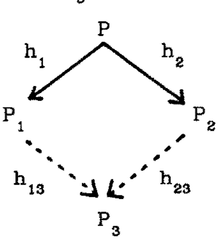
iii) $q \leq p \leq q'$ implies $q \sim p \sim q'$

(*antisymmetry* of $\leq$ is preserved)

It can be shown [C] that there is a one-to-one correspondence between congruences and abstraction homomorphisms on a NDP P: any congruence on P is the *kernel* $\sim_h$ of some a.h. h on P, and any a.h. on P is the *natural mapping* $h_\sim$ associated to some congruence ~ on P. Then the following fact is (almost) standard:

Theorem 3.1 : (*Confluence of abstraction homomorphisms*)

If P, P_1, P_2 are NDP's, and $h_1 : P \longrightarrow P_1$, $h_2 : P \longrightarrow P_2$ are a.h.'s, then $\exists$ NDP P_3, $\exists$ a.h.'s $h_{13} : P_1 \longrightarrow P_3$, $h_{23} : P_2 \longrightarrow P_3$ s.t. the following diagram commutes:



Hint for proof: for the complete proof we refer to [C]. We will just mention here that, if $\sim_1$ and $\sim_2$ are the kernels of h_1 and h_2 , then the a.h. $h_1 h_{13} = h_2 h_{23}$ is the natural mapping associated to the congruence $\sim_3 = [\sim_1 \cup \sim_2]^*$ □

Corollary 3.1: ($\xrightarrow{abs}$ is Church–Rosser)

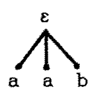
If P, P_1, P_2 are NDP's, and $P_1 \xleftarrow{abs} P \xrightarrow{abs} P_2$

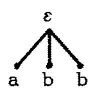
then $\exists$ NDP P_3 s.t. $P_1 \xrightarrow{abs} P_3 \xleftarrow{abs} P_2$

□

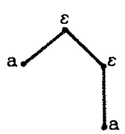
3.1 Abstraction equivalence

The relation $\xrightarrow{abs}$ gives us a criterion to regard two processes as "abstractly the same". However, being essentially a simplification, $\xrightarrow{abs}$ is not symmetric and therefore does not, for example, relate the two processes:





or the processes:



Based on $\xrightarrow{abs}$, we will then define on NDP's a more general relation $\sim_{abs}$, of *reducibility to a same* process:

Definition 3.1.1 : $\sim_{abs} =_{def} \xrightarrow{abs} . \xleftarrow{abs}$

We can immediately prove a few properties for $\sim_{abs}$.

Property 1: $\sim_{abs}$ is an *equivalence.*

Proof: Transitivity follows from the fact that $\xrightarrow{abs}$ is Church–Rosser, which can be restated as: $[\xrightarrow{abs} \cup \xleftarrow{abs}]^* = \xrightarrow{abs} . \xleftarrow{abs}$ □

Property 2: $\sim_{abs}$ is preserved by the operators μ. and +.

Proof: Consequence of $\xrightarrow{abs}$ and $\xrightarrow{abs}^{-1}$ invariance in μ. and + contexts. □

To sum up, we have now a *substitutive* equivalence $\sim_{abs}$ for NDP's that can be split, when required, in two reduction halves. The equivalence $\sim_{abs}$ will be called *abstraction equivalence*. In the coming section we will study how abstraction equivalence relates to bisimulation equivalence, a notion introduced by D. Park [Pa] for general transition systems.

4. Bisimulation relations

A natural method for comparing different systems is to check to which extent they can *behave like* each other, according to some definition of behaviour.

Now, what is to be taken as the *behaviour* of a system need not be known a priori. One can always, in fact, having fixed a criterion for deriving subsystems, let the behaviour of a system be recursively defined in terms of the behaviours of its subsystems.

Based on such an *implicit* notion of behaviour, one gets an (equally implicit) notion of equivalence of behaviour, or *bisimulation*, between systems: two systems are said to bisimulate each other iff any subsystem of either of the two, selected with some criterion, recursively bisimulates a subsystem of the other, selected with the same criterion.

For an NDS S, the transition relation provides an obvious criterion for deriving a subsystem S': S' is a μ-subsystem of S iff $S \xrightarrow{\mu} S'$ for some μ. However, if we are to abstract from internal transitions, a weaker criterion will be needed. To this purpose the following *weak* transition *relations* $\overset{\mu}{\Longrightarrow}$ are introduced:

$$\overset{a}{\Longrightarrow} \; = \; \xrightarrow{\tau}{}^{n} \xrightarrow{a} \xrightarrow{\tau}{}^{m} \qquad\qquad n,\ m \geq 0$$

$$\overset{\tau}{\Longrightarrow} \; = \; \xrightarrow{\tau}{}^{n} \qquad\qquad n \geq 0$$

S' is called a *μ-derivative of S* iff $S \overset{\mu}{\Longrightarrow} S'$. We can then formally define bisimulations on NDS's as follows:

Definition 4.1: A (*weak*) *bisimulation* relation is a relation $R \subseteq (\mathcal{S} \times \mathcal{S})$ such that $R \subseteq F(R)$, where $(S_1, S_2) \in F(R)$ iff $\forall\ \mu \in A$:

i) $S_1 \overset{\mu}{\Longrightarrow} S'_1$ implies $\exists\ S'_2$ s.t. $S_2 \overset{\mu}{\Longrightarrow} S'_2$, $S'_1\ R\ S'_2$

ii) $S_2 \overset{\mu}{\Longrightarrow} S'_2$ implies $\exists\ S'_1$ s.t. $S_1 \overset{\mu}{\Longrightarrow} S'_1$, $S'_1\ R\ S'_2$

Now we know that F has a *maximal fixed-point* (which is also its maximal postfixed-point) given by $\cup_{R \subseteq F(R)}\{R\}$. We will denote this largest bisimulation by $<\approx>$, and, since $<\approx>$ turns out to be an equivalence, refer to it as *the bisimulation equivalence*.

Unfortunately, $<\approx>$ is not preserved by all the operators. Precisely, $<\approx>$ is not preserved by the operator +, as shown by the example:

 $<\approx>$ NIL , but  $<\not\approx>$ 

On the other hand the relation $<\approx>^+$, obtained by closing $<\approx>$ w.r.t. the operator +:

$$S_1 <\approx>^+ S_2 \quad \text{iff } \forall\ S:\ S + S_1 <\approx> S + S_2$$

can be shown to be a *substitutive* equivalence, and in fact to be the largest such equivalence contained in $<\approx>$. (For more details on $<\approx>$ and $<\approx>^+$ we refer to [M2]).

To conclude, $<\approx>^+$ seems a convenient restriction on $<\approx>$ to adopt when modelling NDS's. We will see in the next section that $<\approx>^+$ coincides, on NDP's, with our abstraction equivalence $\sim_{abs}$.

5. Relating Bisimulations to Abstraction Homomorphisms

Looking back at out relations $\xrightarrow{abs}$ and $\sim_{abs}$, we notice that they rely on a notion of *equivalence of states* which, like bisimulations, is *recursive*. Moreover, the recursion builds up on the basis of a similarity requirement (equality of *labels*) that reminds of the criterion (equality of *observable derivation sequences*) used in bisimulations to derive "bisimilar" subsystems. All this indicates there might be a close analogy between abstraction equivalence and bisimulation equivalence.

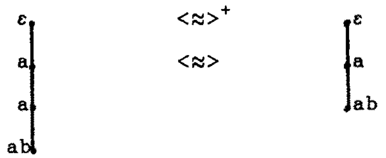
In fact, since we know that $\sim_{abs}$ is substitutive, we shall try to relate it with the substitutive bisimulation equivalence $<\approx>^+$. To this purpose, we will need a direct (recursive) definition for $<\approx>^+$.

Note that $<\approx>^+$ only differs from $<\approx>$ in that it takes into account the *preemptive capacities* a system can develop when placed in a sum-context. Such preemptive capacities depend on the system having some silently reachable state where, informally speaking, some of the "alternatives" offered by the sum-context are no more available. This suggests that we should adopt, when looking for a direct definition of $<\approx>^+$, the more *restrictive* transition *relations* $\stackrel{\mu}{\models\!\!\Longrightarrow}$:

$$\stackrel{\mu}{\models\!\!\Longrightarrow} \; = \; \xrightarrow{\tau}^{n} \xrightarrow{\mu} \xrightarrow{\tau}^{m} \qquad\qquad n,m \geq 0$$

In particular, we will have $\stackrel{\tau}{\models\!\!\Longrightarrow} \; = \; \xrightarrow{\tau}^{n}$, $n>0$. Note on the other hand that, for $a \in A$, it will be: $\stackrel{a}{\models\!\!\Longrightarrow} \; = \; \stackrel{a}{\Longrightarrow}$.

However, $<\approx>^+$ is *restrictive* with respect to $<\approx>$ only as far as the first $\stackrel{\tau}{\Longrightarrow}$ derivation steps are concerned: at further steps $<\approx>^+$ behaves like $<\approx>$, as it can be seen from the example:



So, if we are to recursively define $<\approx>^+$ in terms of the transitions $\stackrel{\mu}{\models\!\!\Longrightarrow}$, we will have to somehow counteract the strengthening effect of the $\stackrel{\mu}{\models\!\!\Longrightarrow}$'s at steps other than the first.

To this end, for any relation $R \subseteq (\mathcal{S} \times \mathcal{S})$, a relation R_a ("almost" R) is introduced: $(S_1, S_2) \in R_a$ iff $(S_1,S_2) \in R$, or $(\tau S_1,S_2) \in R$, or $(S_1,\tau S_2) \in R$

Then we can define a-bisimulation ("almost" bisimulation) relations on NDS's as follows:

Definition 5.1: A (*weak*) *a-bisimulation* relation is a relation $R \subseteq (\mathcal{S} \times \mathcal{S})$ such that $R \subseteq F_a(R)$, where $(S_1,S_2) \in F_a(R)$ iff $\forall\, \mu \in A$:

i) $S_1 \stackrel{\mu}{\models\!\!\Longrightarrow} S_1'$ implies $\exists\, S_2'$ s.t. $S_2 \stackrel{\mu}{\models\!\!\Longrightarrow} S_2'$, $S_1' \, R_a \, S_2'$

ii) $S_2 \stackrel{\mu}{\models\!\!\Longrightarrow} S_2'$ implies $\exists\, S_1'$ s.t. $S_1 \stackrel{\mu}{\models\!\!\Longrightarrow} S_1'$, $S_1' \, R_a \, S_2'$

Again, F_a has a *maximal (post)fixed-point* which is an equivalence, and which we will denote by $<\approx>^a$. The equivalence $<\approx>^a$ has been proven to *coincide* with $<\approx>^+$. Both the definition of $<\approx>^a$ and the proof that $<\approx>^a = <\approx>^+$ are due to M. Hennessy.

It can be easily shown that, if R is an a-bisimulation, then R_a is an ordinary bisimulation. In particular, for the maximal a-bisimulation $<\approx>^a$, it is the case that $<\approx>^a_a = <\approx>$.

Now, it can be proved that:

Theorem 5.1: $\xrightarrow{abs}$ is an a-bisimulation.

The proof relies on the two following lemma's:

Lemma 5.1: If $P_1 \xrightarrow{abs} P_2$ then:

$$P_1 \overset{\mu}{\models\!\!\Longrightarrow} P'_1 \text{ implies } \exists\, P'_2 \text{ s.t. } P_2 \overset{\mu}{\models\!\!\Longrightarrow} P'_2 \text{ where } \textit{either } P'_1 \xrightarrow{abs} P'_2 \textit{ or } P'_1 \xrightarrow{abs} \tau P'_2.$$

Lemma 5.2: If $P_1 \xrightarrow{abs} P_2$ then:

$$P_2 \overset{\mu}{\models\!\!\Longrightarrow} P'_2 \text{ implies } \exists\, P'_1 \text{ s. t. } P_1 \overset{\mu}{\models\!\!\Longrightarrow} P'_1 \text{ where } \textit{either } P'_1 \xrightarrow{abs} P'_2 \textit{ or } P'_1 \xrightarrow{abs} \tau P'_2.$$

Note that in lemma's 5.1 and 5.2 we do not need consider the case $\tau P'_1 \xrightarrow{abs} P'_2$. The reason this case does not arise is that a.h.'s are *single-valued* relations.

Corollary 5.1: $\xrightarrow{abs} \;\subseteq\; <\approx>^a$

Proof: $<\approx>^a$ is the maximal a-bisimulation □

Moreover, we have the following characterisation for a.h.'s:

Terminology: For any NDP P, let $S_P = S_P = \{P' \mid P \longrightarrow^* P'\}$. We say that a bisimulation (a-bisimulation) relation R is *between* P_1 and P_2 iff $(P_1, P_2) \in R$ and $R \subseteq (S_{P_1} \times S_{P_2})$.

Theorem 5.2: An abstraction homomorphism from P_1 to P_2 is a *single-valued* relation which is both a *bisimulation* and an *a-bisimulation* between P_1 and P_2.

We now come to our main result, concerning the relationship between the abstraction equivalence $\sim_{abs}$ and the substitutive bisimulation equivalence $<\approx>^a$. It turns out that these two equivalences coincide:

Theorem 5.3: $\sim_{abs} = <\approx>^a$

Proof of $\subseteq$: From corollary 5.1 we can infer that $\sim_{abs} = [\, \xrightarrow{abs} \xrightarrow{abs}{}^{-1}] \subseteq <\approx>^a$, since $<\approx>^a$ is symmetrically and transitively closed.

Proof of $\supseteq$: Suppose $P_1 <\approx>^a P_2$. We want to show that $\exists\, P_3$ s.t. $P_1 \xrightarrow{abs} P_3 \xleftarrow{abs} P_2$.

Let R be an a-bisimulation between P_1 and P_2. Then R can be written as:

$$R = (P_1, P_2) \cup R\restriction [(S_{P_1} - P_1) \times (S_{P_2} - P_2)]$$

Now consider:

$$R' = (P_1, P_2) \cup R_a\restriction [(S_{P_1} - P_1) \times (S_{P_2} - P_2)]$$

It is easy to see that R' is both a bisimulation and an a-bisimulation between P_1 and P_2. However R' will not, in general, be single-valued. Let then $\sim$ be the equivalence induced by R' on the states of P_2:

$$q_{P'_2} \sim q_{P''_2}$$

iff $\exists\, P'_1 \in S_{P_1}$ s.t. both (P'_1, P'_2) and $(P'_1, P''_2) \in R'$.

It can be shown that $\sim$ is a congruence on P_2 and therefore $\exists\, P_3$ s.t. $h_{\sim}: P_2 \longrightarrow P_3$ is an a.h.. So $P_2 \xrightarrow{abs} P_3$.

Also, by theorem 5.2, $h_{\sim}$ can be regarded as a bisimulation R'' between P_2 and P_3 .Consider now the composition R'·R'': this is by construction a *single-valued* relation contained in $(S_{P_1} \times S_{P_3})$ and containing (P_1, P_3). Moreover R'·R'' is a bisimulation and an a-bisimulation, because both R' and R'' are. So, by theorem 5.2 again, $P_1 \xrightarrow{abs} P_3$.

Summing up, we have $P_1 \xrightarrow{abs} P_3 \xleftarrow{abs} P_2$. □

In view of the last theorem, $\sim_{abs}$ can be regarded as an alternative definition for $<\approx>^a = <\approx>^+$. In the next section, we will see how this new characterisation can be used to derive a set of reduction rules for $<\approx>^+$ on finite processes.

6. A language for finite processes

In this section, we study the subclass of *finite* NDP's, and show how it can be used to model terms of a simple language L.

The language is essentially a subset of R. Milner's CCS (Calculus of Communicating Systems[M1]). In [HM] a set of axioms is presented for L that exactly characterises the equivalence $<\approx>^a$ (and therefore $\sim_{abs}$) on the corresponding transition systems. We show here that the reduction $\xrightarrow{abs}$ itself can be characterised algebraically, by a set of *reduction rules*. These rules yield *normal forms* which coincide with the ones suggested in [HM].

Finally, we establish a notion of *minimality* for NDP's and use it to define a denotational model for L, a class of NDP's that we call *Representation Trees*. The model is shown to be isomorphic with Hennessy and Milner's term-model.

We shall now introduce the language L. Following the approach of [HM], we define L as the term algebra T_Σ over the signature:

$$\Sigma = A \cup \{ NIL, + \}$$

If we assume the operators in Σ to denote the corresponding operators on NDP's (A will denote the set of unary operators μ.), we can use *finite NDP's* to model terms in T_Σ. For a term t, we will use P_t for the corresponding NDP.

We shall point out, however, that the denotations for terms of T_Σ in P will always be *trees*, i.e. NDP's $P = (Q \cup \{r\}, \leq, l)$ obeying the further constraint:

confluence-freeness: $\exists\, q''$ s.t. $q \leq q''$ and $q' \leq q''$

implies $q \leq q'$ or $q' \leq q$

Consider now the set of axioms:

E_c

- sum-laws	E1. $x + x' = x' + x$
	E2. $x + (x' + x'') = (x + x') + x''$
	E3. $x + NIL = x$
- τ-laws	E4. $\mu\tau x = \mu x$
	E5. $\tau x + x = \tau x$
	E6. $\mu\ (x + \tau y) + \mu y = \mu\ (x + \tau y)$
- absorption law	E7. $x + x = x$

Let $=^c$ be the equality generated by E_c. It has been proved [HM] that E_c is a sound and complete axiomatisation for Milner's *observational congruence* $\approx^c$ [M1] , namely that:

$$t =^c t' \text{ iff } P_t \approx^c P_{t'}$$

The relation $\approx^c$ is defined as the closure w.r.t. sum-contexts of the relation (Milner's *observational equivalence*)

$$\approx \ = \ \bigcap_n F^n(P \times P)$$

where $(P \times P)$ is the universal relation on NDP's and F is the function on relations introduced in section 4.

For *image-finite* systems*, the relations $\approx$ and $\approx^c$ have been shown [HM] to coincide with the relations $<\approx>$ and $<\approx>^a$ introduced in the previous sections. In particular, we can assume $\approx^c$ to be defined as $<\approx>^a$ on finite NDP's. Combining these facts together, we have that:

$$t =^c t' \quad \text{iff} \quad P_t \sim_{abs} P_{t'}$$

So $=^c$ is an algebraic analogue for $\sim_{abs}$. Note on the other hand that, although each axiom of E_c could be viewed as a reduction rule (when applied from left to right), the corresponding reduction relation would not characterise $\xrightarrow{abs}$. Consider for example the terms t = aNIL+τ(aNIL+bNIL), t' = τ(aNIL+bNIL). Then the transformation: $t \longrightarrow t'$ would *not* be allowed, whereas we have $P_t \xrightarrow{abs} P_{t'}$.

However, using the axiomatisation E_c as reference, we are able to derive a new system of reduction rules, which characterises $\xrightarrow{abs}$.

We first need to define the relations $\xrightarrow{\mu}$ on terms of T_Σ: $\forall\ \mu \in A^*$, $\xrightarrow{\mu}$ is the *least* relation satisfying the rules:

i) $\mu t \xrightarrow{\mu} t$

ii) $t \xrightarrow{\mu} t'$ implies $t + t'' \xrightarrow{\mu} t'$, $t'' + t \xrightarrow{\mu} t'$

---

*Our restriction on the labelling for NDP's corresponds to the *general image-finiteness* condition: $\forall q, \forall \mu, \{q' \mid q \overset{\mu}{\Rightarrow} q'\}$ is finite

The weak relations $\stackrel{\mu}{\Longrightarrow}$ are derived from the $\stackrel{\mu}{\longrightarrow}$'s just as in section 4.

Let now $\longrightarrow^c$ be the reduction relation generated by the following set of reduction rules R_c (where $\longleftrightarrow$ stands for $(\longrightarrow \cap \longrightarrow^{-1})$):

R_c

	R1.	$x + x' \longleftrightarrow x' + x$
- sumlaws	R2.	$(x + x') + x'' \longleftrightarrow x + (x' + x'')$
	R3.	$x + NIL \longrightarrow x$
- 1st τ-law	R4.	$\mu\tau x \longrightarrow \mu x$
- generalised absorption law	R5.	$x + \mu x' \longrightarrow x$, whenever $x \stackrel{\mu}{\Longrightarrow} x'$

Then it can be proved [C] that:

Theorem 6.1: $t \longrightarrow^c t'$ iff $P_t \stackrel{abs}{\Longrightarrow} P_{t'}$

Corollary 6.1: R_c is a rewriting system for the equational theory E_c.

We can make use of our new axiomatisation for $=^c$ to characterise normal forms for terms in T_Σ. We say that a term is in *normal form* if no proper reduction (R3, R4 or R5) can be applied to it. It can be shown that:

Theorem 6.2: A term $t = \Sigma_i\ \mu_i t_i$ is a *normal form* iff (Hennessy-Milner characterisation):

i) no t_i is of the form $\tau t'$

ii) each t_i is a normal form

iii) for $i \neq j$, $t_i \longleftrightarrow t'_j\ \ \forall\, t'_j$ s.t. $\mu_j t_j \stackrel{\mu_i}{\Longrightarrow} t'_j$

Corresponding to normal forms, we have a notion of *minimality* for processes. We say that a process P is *irreducible* or *minimal* iff $P \stackrel{abs}{\Longrightarrow} P'$ implies $P = P'$. Then the following is trivial:

Theorem 6.3: For any finite NDP P, $\exists\,!$ *minimal* NDP P' s.t. $P \sim_{abs} P'$.

Proof: for uniqueness, use $\stackrel{abs}{\Longrightarrow}$'s Church-Rosser property □

We shall denote by $\hat{P}$ the unique minimal process corresponding to the NDP P.

Corollary 6.2: $P \sim_{abs} P'$ iff $\hat{P} = \hat{P}'$. □

As we mentioned earlier, the denotation P_t of a term t is always a *tree*. However its "abstract" denotation $\hat{P}_t$ might not be a tree. We shall now propose a *tree-model* for terms of T_Σ, which is isomorphic to the term-model $T_\Sigma/=^c$.

Note first that any NDP which is not a tree has a *unique unwinding* into a tree. The tree-unwinding of an NDP P (which is not defined formally here) will be denoted by U(P).

Let now *RT* (*representation trees*) be the class: RT = { U(P) | P is a minimal NDP }.The *denotation* T_t of a term $t \in T_\Sigma$ in RT is defined by: $T_t = U(\hat{P}_t)$.

It can be shown that:

Theorem 6.4: $t =^c t'$ iff $T_t = T_{t'}$ □

We shall finally argue that our model *RT* is *isomorphic* to the term-model $T_\Sigma/=^c$:

RT is a Σ-algebra satisfying the axioms E_c (by theorem 6.4), with the operators defined by:

$$\mu U(P) = U(\widehat{\mu P})$$

$$U(P_1) + U(P_2) = U(\widehat{P_1 + P_2})$$

Therefore, since $T_\Sigma/=^c$ is the initial Σ-algebra satisfying the axioms E_c, we know that:

$$\exists\ !\ \ \Sigma\text{-homomorphism}\ \ \Psi : T_\Sigma/=^c \longrightarrow RT$$

It is easily seen that Ψ is given by: $\Psi([t]) = U(\hat{P}_t) = T_t$. Also, by theorem 6.4 again, Ψ is a *bijection* between T_Σ and *RT*.

Conclusion

We have proposed an alternative definition for the (substitutive) bisimulation equivalence $<\approx>^+$ for a class of transition systems. Note that the ordinary bisimulation equivalence could be characterised just as easily, by slightly changing the definition of homomorphism: in fact it would be enough to drop the requirement that *proper* states should be preserved. Also, using our definition, we have been able to derive a denotational model for the language L, which is isomorphic to Hennessy and Milner's term model for the same language.

Our approach is intended to extend to richer languages, for programs which are *both* nondeterministic *and* concurrent (meaning that the actual concurrency is not interpreted nondeterministically). Some simple results have already been reached in that direction.

Acknowledgements

The definition of abstraction homomorphism and the idea of using it to characterise Milner's notions of observational equivalence and congruence stems from a joint work with U. Montanari at Pisa University. I would like to thank him for inspiration and for subsequent discussions. I would also like to thank my supervisor M. Hennessy for the substantial help he gave me all along, and R. Milner for helpful suggestions. Many thanks to my colleagues Francis Wai and Tatsuya Hagino for helping me with the wordprocessing of the paper.

References

LNCS stands for Lecture Notes in Computer Science, Springer-Verlag

[BR] S. Brookes, C. Rounds (1983), "Behavioural Equivalence Relations induced by Program Logics", in Proc. ICALP '83, LNCS 154.

[C] Full version of this paper. Contact the author.

[CFM] I. Castellani, P. Franceschi, U. Montanari (1982), "Labelled Event Structures: A Model for Observable Concurrency", in: D. Bjorner (ed.):Proc. IFIP TC2 Working Conference on Formal Description of Programming Concepts II, Garmisch, June 1982: North-Holland Publ. Company 1983

[DeN] R. De Nicola (1984), "Behavioural Equivalences for Transition Systems", Internal Report I.E.I., Pisa, Italy.

[HM] M. Hennessy, R. Milner (1983), "Algebraic laws for Nondeterminism and Concurrency" , Technical Report: CSR-133-83 , University of Edinburgh.

[K] R. Keller (1976), "Formal verification of Parallel Programs", Communications of the ACM n. 19, Vol. 7.

[M1] R. Milner (1980), A Calculus of Communicating Systems, LNCS 92.

[M2] R. Milner (1982), "Calculi for Synchrony and Asynchrony", J. Theoretical Computer Science, Vol. 25.

[Pa] D. Park (1981), "Concurrency and Automata on Infinite Sequences", in LNCS 104.

[P] G. Plotkin (1981), "A Structured Approach to Operational Semantics", DAIMI FN-19, Computer Science Dept, Aarhus University.