ISSN 1361 - 6161

THE UNIVERSITY of MANCHESTER

Computer Science
University of Manchester

Modal Observation Equivalence of Processes

W. P. R. Mitchell and D. P. Carlisle

Department of Computer Science

University of Manchester

Technical Report Series

UMCS-96-1-1

Modal Observation Equivalence of Processes*

W. P. R. Mitchell, D. P. Carlisle

Department of Computer Science
University of Manchester
Oxford Road, Manchester, UK.
{bill, carlisle}@cs.man.ac.uk

30-3-95

*Refereed by: Dr C. D. Tofts

Abstract

This paper shows that many simulation equivalences over processes can be characterised by a single parameterised modal logic formula. For many equivalences $\sim_\sigma$ there is a parameterised modal logic formula $f(t)$ such that for every pair of processes P and Q, $P \sim_\sigma Q$ if and only if $P \models f(Q)$. For any specific P the formula $f(P)$ becomes a fix point formula of the normal modal logic $\mu\mathbb{K}$.

Strong bisimulation equivalence $\sim$ was shown by Milner [16] to be characterised in this way. This paper shows that simulation, complete simulation, ready simulation, 2-nested simulation, weak bisimulation and congruence can also be characterised in this way.

The paper constructs a template which can be used to generate a parameterised modal formula which is guaranteed to characterise a 'sensible' simulation equivalence. Two infinite families of simulation equivalences are characterised in this way, both of which converge to $\sim$. The paper defines a similar template for weak simulation equivalences. This template is used to construct an infinite family of equivalences which converges to weak bisimulation equivalence, and another infinite family which converges to congruence.

Any equivalence relation which contains $\sim$ can be characterised in a weaker sense by $\mu\mathbb{K}$. If $\sim_\sigma$ contains $\sim$, then for each P there is an $\mathsf{f}^\sigma_P \in \mu\mathbb{K}$ such that $P \sim_\sigma Q$ iff $\models \mathsf{f}^\sigma_P \leftrightarrow \mathsf{f}^\sigma_Q$.

The paper proves this for arbitrary σ via the axiom of choice. The paper gives a specific construction for such an f^σ_P for trace, complete trace, failure trace, ready trace and possible future equivalence .

1 Introduction

There are over a hundred different observation equivalence relations now listed in the literature. These equivalences range from trace equivalence at one extreme to strong bisimulation at the other. Van Glabbeek [6], [7] has produced some excellent surveys of most of these equivalences. Essentially an observation equivalence $\sim_\sigma$ is any equivalence over process terms which contains $\sim$, strong bisimulation equivalence [16]. That is, $P \sim Q \Rightarrow P \sim_\sigma Q$.

This paper demonstrates that there is a unified way of giving semantics to these equivalences in terms of a single modal logic $\mu\mathbb{K}$ (for Kripke). This is a particularly simple logic. It is the logic $\mathbb{K}$ (see [15] for details) with an added fix point operator μ. The only modal operators for $\mu\mathbb{K}$ are $\Box$ always and $\Diamond$ sometime. The formula $\Box f$ loosely translates as: in every next state from here, f is true; the formula $\Diamond f$ translates as: in some next state from here, f is true.

At the weakest level we can say the following about any observational equivalence $\sim_\sigma$, for each image finite process P there is a formula $\mathsf{f}^\sigma_P \in \mu\mathbb{K}$ such that for every P and Q, $P \sim_\sigma Q$ if and only if $\models \mathsf{f}^\sigma_P \leftrightarrow \mathsf{f}^\sigma_Q$. Such an equivalence σ is called a *logic equivalence*.

It is remarkably simple to prove this result. Using the axiom of choice pick one process P_σ from each $\sim_\sigma$ equivalence class $[P]$. For any process P there is a well defined modal logic formula (given precisely in a later section) $\mathsf{f}_\sim(P)$ such that $P \sim Q$ if and only if $\models \mathsf{f}_\sim(P) \leftrightarrow \mathsf{f}_\sim(Q)$. Clearly for any P, Q we know that $P \sim_\sigma Q$ if and only if $P_\sigma \sim Q_\sigma$, if and only if $\models \mathsf{f}_\sim(P_\sigma) \leftrightarrow \mathsf{f}_\sim(Q_\sigma)$. Therefore one can define $\mathsf{f}^\sigma_P = \mathsf{f}_\sim(P_\sigma)$.

A multi-modal logic formula $\mathsf{f}_\sim$ characterising $\sim$ as above has been known for some time, see [16] or [25]. The above argument therefore works equally well in constructing a multi-modal characterisation of $\sim_\sigma$. One of the aims in this paper is to show that there is a simple logical framework in which to place observational equivalences. The logic $\mu\mathbb{K}$ is used here because it is much simpler than multi-modal logic. It has only two modalities, as opposed to a larger number in multi-modal logic, and the Kripke models which define satisfiability are simple and do not require the idea of multiple time frames which are required in multi-modal logic.

The axiom of choice is not much help in constructing any specific value for f_P^σ. Nor does an existence proof provide any insight into how concise the formula f_p^σ could be made or how computationally effective such a characterisation might be. This paper gives an algorithmic description of a formula f_P^σ for trace, complete trace, failure trace, ready trace and possible future equivalence. Each of these formulae are concise in terms of the size of the original process, and give a decision procedure for $\sim_\sigma$ which is as computationally efficient as the original definition.

For equivalences which have a deeper structure than the trace like equivalences mentioned above, there is a much stronger $\mu\mathbb{K}$ characterisation, which is described below, after first introducing some notation.

One can extend $\mu\mathbb{K}$ slightly to define a predicate modal logic $\mathbb{K}^{Par}$ which allows formulae to be parameterised by image finite processes. Formulae of $\mathbb{K}^{Par}$ are monadic, that is they have at most one parameter. Given any formula f of $\mathbb{K}^{Par}$ and an image finite process x, then $f(x)$ always evaluates to a formula of $\mu\mathbb{K}$.

An image finite process, by definition, is determined by a finite transition system, which in turn can be depicted by a finite edge labeled graph with a distinguished start node.

There is a simple trick which allows a finite edge labelled graph to be identified with a Kripke model for $\mu\mathbb{K}$. Let $K(P)$ be the Kripke model which is generated by the finite graph depicting P. This is defined precisely in the next section, see definition 2. For a modal formula f and process P we write $P \models f$ to mean $K(P) \models f$.

An equivalence relation $\sim_\sigma$ is a *modal equivalence* if there is a formula $\mathsf{f}_\sigma \in \mathbb{K}^{Par}$ such that for every P and Q,

$$P \sim_\sigma Q \Leftrightarrow P \models \mathsf{f}_\sigma(Q)$$

The formula f_σ is the *characteristic formula* for σ. Since $\sim_\sigma$ is an equivalence relation, and therefore symmetric, it follows that $P \sim_\sigma Q \Leftrightarrow Q \models \mathsf{f}_\sigma(P)$. In the paper simulation, complete simulation, ready simulation, 2-nested simulation, weak bisimulation, congruence and strong bisimulation are proved to be modal equivalences. Moreover the formula f_σ in each of these cases also makes σ a logic equivalence.

One important point about a modal equivalence σ is that the characteristic formula $\mathsf{f}_\sigma(t)$ always evaluates to a $\mu\mathbb{K}$ formula for every possible value P of its parameter t. Hence the semantics of a modal equivalence are embedded within $\mu\mathbb{K}$, which is a particularly simple logic.

For a process term P of maximum fan out n and depth m, all of the formulae $\mathsf{f}_\sigma(P)$, for the various bisimulation equivalences, are of size $O(nm)$. For simulation, complete simulation, ready simulation and 2-nested simulation $\mathsf{f}_\sigma(P)$ is linear in the length of P. Model checking in $\mu\mathbb{K}$ is very effective when measured in terms of the size of model and formula, so that the decision procedure for $\sim_\sigma$ when translated into modal logic looses none of its efficiency.

For each of the particular trace like equivalences mentioned above none of them appear to be modal equivalences. If $\mathbb{K}^{Par}$ is extended to allow a formula to have more than one parameter then it is possible to characterise the various trace like equivalences as modal equivalences over the extended logic. However, once more than one parameter is allowed it is not always the case that evaluation of a logic formula at given parameters is a $\mu\mathbb{K}$ formula, which negates the purpose of the exercise.

The paper gives a template for defining a formula $f \in \mathbb{K}^{Par}$ such that the equivalence $\sigma(f)$, defined by $P \sim_{\sigma(f)} Q$ if and only if $P \models f(Q)$, is a 'sensible' simulation equivalence of processes[1].

[1]The term sensible is described precisely in section 3.8

By definition such an equivalence is a modal equivalence. The paper defines two distinct infinite families of simulations based on this template, both of which converge to $\sim$. The paper also defines an infinite family of modal equivalences which converge to weak bisimulation equivalence (denoted $\approx$), and an infinite family of modal equivalences which converge to congruence (denoted $=_\tau$).

These infinite families demonstrate that there is a very rich world of modal equivalences, that this world contains infinite variation, but also contains all the well known observational equivalences. Moreover the various well known bisimulation equivalences are direct limits of the more straightforward simulation like equivalences.

1.1 Related Work

The Hennessey Milner logic [11] was the first attempt to understand bisimulation equivalence in terms of a modal style logic. The essentials of this approach are roughly this. Each process term of CCS can be regarded as a finite edge labeled directed graph. Each edge is labeled with an action name. Some node λ of each graph is designated as a start node.

Introduce multi-modal operators $[a]$ and $\langle a \rangle$. Formula $[a]f$ is true at a node u of a graph G exactly when at every node v, which can be reached from u by an edge labeled with a, the formula f holds true. That is $(G, u) \models [a]f$ if $\forall\ u \xrightarrow{a} v$, $(G, v) \models f$. The operator $\langle a \rangle$ has the semantics: $(G, v) \models \langle a \rangle f$ if and only if $\exists\ u \xrightarrow{a} v$, $(G, v) \models f$.

As well as these modalities the logic has constants True and False, and the usual logical connectives $\wedge$, $\vee$ and $\neg$. For a process P depicted by graph G with start node λ, define $P \models f$ to denote that $(G, \lambda) \models f$.

Hennessey and Milner [11] proved that $P \sim Q$ iff for every Hennessey Milner multi-modal formula f, $P \models f \Leftrightarrow Q \models f$. This formulation has been refined by adding fix point operators to the logic. In doing this it is possible to show (see [25]) that for each process P there is a fix-point multi-modal logic formula $\mathsf{f}(P)$ such that $P \sim Q$ if and only if $P \models \mathsf{f}(Q)$.

Graf and Sifakis [5] defined an alternative multi-modal logic to that of Hennessey and Milner which characterised $\sim$ and simulation equivalence exactly in the same style as Hennessey and Milner. The advantage of Graf and Sifakis' logic was that it provided a much more concise formula for $\mathsf{f}(P)$, and the logic was able to more directly express properties of nondeterminism.

Later Hennessey and Milner [12] showed that if one defines a logic S whose formulae are constructed solely from $[a]$, $\langle a \rangle$, $\wedge$, and constants True and False, then P is simulation equivalent to Q if and only if for every formulae f of S, $P \models f \Leftrightarrow Q \models f$. By allowing fix-point operators in S one can find a formula $\mathsf{f}(P)$ for each process P such that P and Q are simulation equivalent if and only if $P \models \mathsf{f}(Q)$ and $Q \models \mathsf{f}(P)$.

In van Glabbeeks paper [7] there are well over a hundred observational equivalences. The semantics of all these is roughly a generalisation of the above idea of Hennessey and Milner in characterising simulation.

For a particular equivalence $\sim_\sigma$ define various multi-modal operators with which to construct logic formulae; thus defining a multi-modal logic L_σ. Two processes P and Q are then σ equivalent if and only if $\forall f \in L_\sigma$, $P \models f \Leftrightarrow Q \models f$. All these equivalences contain bisimulation equivalence and are therefore logic equivalences as proved above. By giving each equivalence an accompanying multi-modal logic van Glabbeek is able to construct an inclusion lattice hierarchy for the equivalences.

Goltz, Kuiper, and Penczek [8] have taken this approach a step further by looking at logics for L_σ which are not multi-modal but instead work with modalities which are more in character with CTL* like operators. Their paper deals with observations which do not always contain $\sim$ and so fall outside of the scope of this paper.

The π-calculus is beginning to succumb to a similar approach [13]. Hennessey and Liu have described a first order multi-modal logic FL which can be used to characterise strong late bisimulation equivalence in the sense that; for two π-calculus processes P and Q, $P \sim Q$ if and only if for all formulae $f \in FL$, $P \models f \Leftrightarrow Q \models f$. Mads Dam has also published [4] a paper describing a first order logic which characterises strong late bisimulation.

The current state of the literature suggests that there is a proliferation of modal logics, each new equivalence generating a new logic to characterise it. It also appears to be common to identify a process with the set of formulae in a particular logic that the process satisfies. So for a logic L_σ, process P is identified with the set $L_\sigma(P) = \{f \in L_\sigma \mid P \models f\}$, then $P \sim_\sigma Q$ iff $L_\sigma(P) = L_\sigma(Q)$.

The approach of this paper is that for each $\sim_\sigma$ and each process P there should be a single formula $\mathsf{f}^\sigma_P \in \mu\mathbb{K}$ such that $P \sim_\sigma Q$ iff $\mathsf{f}^\sigma_P \leftrightarrow \mathsf{f}^\sigma_Q$. Preferably there is a single formula $\mathsf{f}_\sigma \in \mathbb{K}^{Par}$ such that $P \sim_\sigma Q$ iff $P \models \mathsf{f}_\sigma(Q)$.

1.2 The rest of the paper

The next section defines precisely the modal logic $\mathbb{K}^{Par}$. Section 3 defines logic and modal simulations, which turn out to include simulation, complete simulation, ready simulation, and 2-nested simulation. This class also includes the various infinite families defined in section 4. Finally in section 5 the trace like equivalences mentioned at the beginning of the introduction are constructively shown to be logic equivalences.

2 The logic $\mathbb{K}^{Par}$

Throughout this paper we regard a process as an image finite transition system. Every transition system can be depicted as a finite edge labeled graph with distinguished start node. We take the set of possible labels to come from the set A, the set of action names.

This section defines the algebraic notation used to describe finite graphs. This notation is essentially that of regular expressions, modified slightly to allow fix point notation which is more compatible with modal fix point operators. Also in this section the logics $\mu\mathbb{K}$ and $\mathbb{K}^{Par}$ are formally defined. In section 3 we will prove that various simulation like equivalences are modal equivalences as defined in the introduction.

2.1 Regular expressions

Let A be a set of action names. All processes can perform actions only from the set A. The set of *regular terms* is defined by the BNF equation:

$$T = a \cdot T \mid T + T \mid 0 \mid \nu X.T \mid X$$

where $a \in A$ and X is a meta variable used to construct a cycle. Meta variables may not appear in a subterm of the form $X \cdot T$, so that meta variables range over processes and not actions.

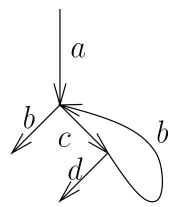
A regular term is a *regular expression* if each meta variable X is bound within the scope of νX. This set of regular expressions can be identified with the set of finite edge labeled directed graphs with a distinguished start node.

For a node u in a graph G and a term t define the graph $G(t, u)$ recursively as follows:

- If $t = 0$ then $G(t, u) = G$.
- If t is of the form $\nu X.s$ then 'mark' u with a label X and generate the graph for $G(s, u)$.
- If t is of the form $\sum_i a_i \cdot t_i$ then for each t_i extend the graph as follows. If t_i is a meta variable, X, add an edge $u \xrightarrow{a_i} v$ where v is the most immediate ancestor node labeled by X. Otherwise add a new child node u_i to u plus new edges $u \xrightarrow{a_i} u_i$. Then recursively construct the subgraph $G(t_i, u_i)$.

When G has a distinguished start node λ let $G(t)$ denote $G(t, \lambda)$. Let $\emptyset$ denote the graph with one node and no edges. One may identify the regular expression t with the graph $\emptyset(t)$.

For example we may identify $a \cdot \nu X(b \cdot 0 + c \cdot (d \cdot 0 + b \cdot X))$ with the graph



Throughout the paper a process is identified with the regular expression which describes the transition system of the process.

2.2 The modal logic $\mu\mathbb{K}$

A formula in $\mu\mathbb{K}$ can be constructed from the logical connectives $\wedge$, $\vee$, $\neg$ and propositional variables from the set A, together with the modal operators $\Box$ and $\Diamond$ and the fix point operator μ. Meta variables (always denoted by upper case letters, e.g. X) are used to define fix point formulae. An *expression* of $\mu\mathbb{K}$ is defined by the BNF formula

$$E = a \mid X \mid E \wedge E \mid E \vee E \mid \neg E \mid \Box\, E \mid \Diamond\, E \mid \mu X.E$$

Where $a \in A$ and X is a meta variable. An expression E is a *formula* if every meta variable X occurs within the scope of an enclosing fix point operator μX. For an expression $E(X)$ where X occurs free in E and an expression x, let $E^n(x)$ denote $E(E(\cdots E(x)\cdots)$, that is E applied n times to x.

A Kripke model of $\mu\mathbb{K}$ is a finite directed graph G whose nodes are labeled by subsets of A. The label of node u is denoted $L(u)$. Write $u \longrightarrow v \in G$ to denote that there are nodes u and v of G and a directed edge from u to v. Some node of G is designated as the start node, this is denoted λ.

The satisfiability relation $\models$ is defined recursively as follows: Let u be a node of graph G,

- $(G, u) \models f \wedge g \Leftrightarrow ((G, u) \models f) \wedge ((G, u) \models g)$
- $(G, u) \models f \vee g \Leftrightarrow ((G, u) \models f) \vee ((G, u) \models g)$
- $(G, u) \models \neg f \Leftrightarrow (G, u) \not\models f$
- $(G, u) \models \Box\, f \Leftrightarrow \forall\, u \longrightarrow v \in G\ .(G, v) \models f$
- $(G, u) \models \Diamond\, f \Leftrightarrow \exists\, u \longrightarrow v \in G\ .(G, v) \models f$

- $(G, u) \models a$ for $a \in A$ iff $a \in L(u)$.
- $(G, u) \models \mu X.E(X)$, where $E(X)$ is an expression where X is the only free meta variable in E, if there is some n such that $(G, u) \models E^n(\mathsf{F})$.

The operator μ is known as the least fix point operator. The maximal fix point operator ν is defined by $\nu X.E = \neg\mu X.\neg E$. The semantics of ν can be given directly as

- $(G, u) \models \nu X.E(X)$, where $E(X)$ is an expression and X is the only free meta variable in E, if for every n, $(G, u) \models E^n(\mathsf{T})$

Definition 1.
Define $G \models f$ iff $(G, \lambda) \models f$.

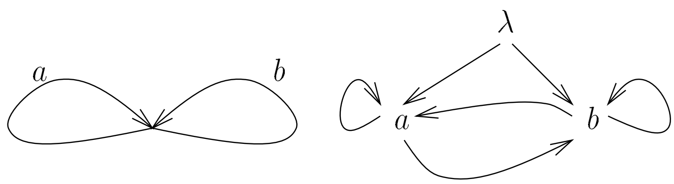
Every image finite process P of CCS can be represented by a finite directed graph R_P with edges labeled by action names of A. This is the transition system defined by P.

Let $G = (V, E)$ be a directed graph with edges labeled by a function $l : E \longrightarrow A$. Define $K(G)$ to be a node labeled graph with node set $E \cup \{\lambda\}$ where $\lambda \notin E$. There is a directed edge in $K(G)$ from $e_1 \in E$ to $e_2 \in E$ iff the end node of e_1 is the start node of e_2. Node e of $K(G)$ is labeled by $l(e)$. There is also an edge in $K(G)$ from λ to each $e \in E$ emanating from the start node of G. Node λ is the start node for $K(G)$ which is labeled with the empty set.

The graph $K(G)$ can be regarded as a Kripke model, where at node u the proposition $l(u)$ is given value True and all other propositions are made false. Thus at λ all propositional variables are set false.

Let $d(u)$ be the degree of a node u, that is the number of edges incident on u. Graph $K(G)$ has approximately $\sum_{x \in E} d(x)$ edges. In the case of a finite process P, which can be depicted as an edge labeled tree, $K(P)$ can be derived by simply 'sliding' each edge label along it's arrow until it lands on a node. In this case $K(P)$ is exactly the same size as P. If P were strongly connected, so there are paths between any two nodes, then $K(P)$ would have approximately twice as many edges as P.

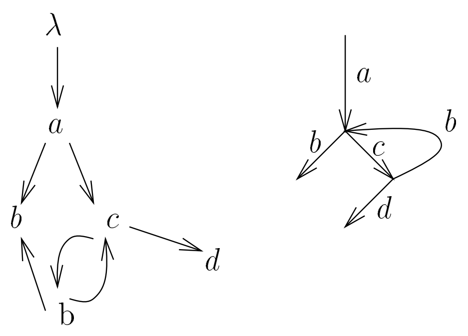
For example



G $K(G)$

Definition 2.
Let P be an image finite process depicted by the finite edge label graph R_P. Define the kripke model $K(P)$ to be the model $K(R_P)$. Define $P \models f$ iff $K(P) \models f$.

For example, take $P = a \cdot \nu X(b \cdot 0 + c \cdot (d \cdot 0 + b \cdot X))$, then we have the graph of the regular expression for P, and the Kripke model for P as depicted below:



$K(P)$ $\qquad\qquad$ P

In this case

$$P \models \Box(a \wedge \nu X.(\Diamond\, b \wedge \Diamond(c \wedge \Diamond\, d \wedge \Diamond(b \wedge X))))$$

2.3 The parameterised modal logic $\mathbb{K}^{Par}$

The formulae of $\mathbb{K}^{Par}$ include those of $\mu\mathbb{K}$. In addition there is a set of unary predicate symbols F so that an expression of $\mathbb{K}^{Par}$ is of the form given by

$$E = \mathsf{f}(P) \mid a \mid X \mid E \wedge E \mid E \vee E \mid \neg E \mid \Box\, E \mid \Diamond\, E \mid \mu X.E$$

Where $\mathsf{f} \in F$ and P is a regular expression/process.

A *formula* of $\mathbb{K}^{Par}$ is just as for $\mu\mathbb{K}$ except that a formula ϕ may also contain predicates $\mathsf{f}(P)$, where $\mathsf{f} \in F$ and P is a regular expression, provided that each such predicate is accompanied by a set of equations of the following form.

$$\begin{array}{lcl} \mathsf{f}(t+s) & = & E_1(t,s) \\ \mathsf{f}(a\cdot t) & = & E_2(a,t) \\ \mathsf{f}(\nu X.t) & = & E_3(t) \\ \mathsf{f}(X) & = & E_4 \\ \mathsf{f}(0) & = & E_5 \end{array}$$

Each E_i is an expression where any predicate other that f occurring in E_i is defined by a set of recursive equations in the same format as those above. Also variables t and s may only occur in a subterm of the form $g(t)$ or $g(s)$, where $g \in F$. Further in E_3 each occurrence of t must be within the scope of a fix point operator νX or μX. In E_4 the variable X may occur as a free logic meta variable, but may not occur in the scope of any $g \in F$. This ensures that, for a given concrete value of P, $\mathsf{f}(P)$ always evaluates to a fix point formula in $\mu\mathbb{K}$.

For a model G the predicates of F have exactly the semantics one would expect. For a regular expression P define $G \models \mathsf{f}(P)$ iff $G \models E_i$ where E_i is the expression belonging to the relevant clause for P in the definition of f.

Here is an example:

$$\begin{array}{lcl} f(t+s) & = & f(t) \wedge f(s) \\ f(a\cdot t) & = & \Diamond(a \wedge f(t)) \\ f(\nu X.t) & = & \nu X.f(t) \\ f(X) & = & X \\ f(0) & = & \Box\,\mathsf{F} \end{array}$$

With this definition

$$
\begin{aligned}
f(a \cdot (b \cdot 0 + c \cdot 0)) &= \Diamond(a \wedge f(b \cdot 0 + c \cdot 0)) \\
&= \Diamond(a \wedge f(b \cdot 0) \wedge f(c \cdot 0)) \\
&= \Diamond(a \wedge \Diamond(b \wedge f(0)) \wedge \Diamond(c \wedge f(0))) \\
&= \Diamond(a \wedge \Diamond(b \wedge \Box\, \mathsf{F}) \wedge \Diamond(c \wedge \Box\, \mathsf{F}))
\end{aligned}
$$

Similarly

$$f(a \cdot \nu X.(bX + cX)) = \Diamond(a \wedge \nu X.((\Diamond\, b \wedge X) \wedge (\Diamond\, c \wedge X)))$$

Notice that $a \cdot \nu X.(bX + cX) \models f(a \cdot \nu X.(bX + cX))$.

2.4 Abbreviated Equations

The notation for defining predicates of $\mathbb{K}^{Par}$ is straightforward but tedious. In all the cases considered in this paper a restricted version of the definition will do. The following notation allows for a restricted form which is much more concise than the full version.

For a set of logic formulae S write $\bigvee S$ to denote $\bigvee_{x \in S} x$, and $\bigwedge S$ to denote $\bigwedge_{x \in S} x$. Let $\cong$ be the equivalence relation on regular expressions generated by the equations $t + s \cong s + t$ and $t + (s + r) \cong (t + s) + r$. Let $P \overset{a}{\longmapsto} P'$ denote that there is a regular expression P'' such that $P \cong a \cdot P' + P''$.

Let Eq be the set of equations:

- $\mathsf{f}(P) = E(\bigvee\{E'(a, P') \mid P \overset{a}{\longmapsto} P'\})$
- $\mathsf{f}(0) = E_1$

Any predicate $g \in F$ which appears in $E(t)$ or $E'(a, t)$ must be defined by a full set of equations if $g \neq \mathsf{f}$, also t must only occur in subterms of the form $g(t)$ for some $g \in F$. Also $E'(a, t)$ is an expression where a only appears as a propositional variable.

The intended meaning for these equations is that f performs the obvious expansion on fix points, i.e. $\mathsf{f}(\nu X.t) = \nu X.\mathsf{f}(t)$ and $\mathsf{f}(X) = X$. The first equation is intended to explain how f recurses over an expression which is a sum of terms. Thus, if $P = \sum_i a_i \cdot P_i$ the first equation performs the expansion $\mathsf{f}(P) = E(\bigvee_i E'(a_i, P_i))$. More formally, this set Eq is an abbreviation for the equations

- $\mathsf{f}(t + s) = E(f_1(t) \vee f_1(s))$
- $\mathsf{f}(a \cdot P) = E(E'(a, P))$
- $\mathsf{f}(\nu X.t) = \nu X.\mathsf{f}(t)$
- $\mathsf{f}(X) = X$
- $\mathsf{f}(0) = E_1$
- $f_1(t + s) = f_1(t) \vee f_1(s)$
- $f_1(a \cdot P) = E'(a, P)$

- $f_1(\nu X.t) = \nu X.f_1(t)$
- $\mathsf{f}(X) = X$.
- $f_1(0) = E_1$

The function f_1 is just an auxiliary function to allow f to expand properly over a sum of terms. For example consider the predicate defined by

- $\mathsf{f}(P) = \Box(\bigvee\{(\Diamond \neg a) \wedge \mathsf{f}(P') \mid P \overset{a}{\longmapsto} P'\})$
- $\mathsf{f}(0) = \Diamond \mathsf{F}$

Here is an explicit example of evaluation of f on the expression $\nu X.(a \cdot (b \cdot X + c \cdot 0))$

$$\begin{aligned}
&\mathsf{f}\big(\nu X.(a \cdot (b \cdot X + c \cdot 0))\big)\\
&= \nu X.\mathsf{f}\big(a \cdot (b \cdot X + c \cdot 0)\big)\\
&= \nu X.\Box(\bigvee\{\Diamond \neg a \wedge \mathsf{f}\big(b \cdot X + c \cdot 0\big)\})\\
&= \nu X.\Box(\Diamond \neg a \wedge \Box(\bigvee\{\Diamond \neg b \wedge \mathsf{f}(X),\ \Diamond \neg c \wedge \mathsf{f}(0)\}))\\
&= \nu X.\Box(\Diamond \neg a \wedge \Box(\Diamond \neg b \wedge X \vee \Diamond \neg c \wedge \Diamond \mathsf{F}))
\end{aligned}$$

The other abbreviation which is allowed is a definition in the form

- $\mathsf{f}(P) = E(\bigwedge\{E'(a, P') \mid P \overset{a}{\longmapsto} P'\})$
- $\mathsf{f}(0) = E_1$

This comes with the same set of conditions as the previous abbreviation, and has the analogous expansion.

3 Modal Simulation

This section defines a class of equivalences in terms of the logic $\mathbb{K}^{Par}$. Each of simulation, complete simulation, ready simulation, 2-nested simulation, weak bisimulation and congruence are proved to be modal equivalences.

For processes P and Q we write $P \xrightarrow{a} Q$ to denote that P can perform action a and evolve into process Q. In terms of graph representation $P \xrightarrow{a} Q$ denotes that there is an edge $\lambda \xrightarrow{a} \lambda_1 \in R_P$, such that the graph R_Q can be identified with the subgraph of R_P descendant from λ_1 in such a way that the start node of R_Q is identified with λ_1.

Definition 3.
An equivalence relation $\sim_\sigma$ over processes is a modal equivalence if there is a formula $\mathsf{f}_\sigma \in \mathbb{K}^{Par}$ such that

$$P \sim_\sigma Q \text{ iff } P \models \mathsf{f}_\sigma(Q)$$

Further $\sim_\sigma$ is a modal simulation if

$$P \models \mathsf{f}_\sigma(Q) \Rightarrow \forall\, P \xrightarrow{a} P'\ \exists\, Q \xrightarrow{a} Q' \text{ such that } P' \models \mathsf{f}_\sigma(Q')$$

Let R be a binary infix relation over processes and $g \in \mathbb{K}^{Par}$. Define R to be a forward g simulation relation if $PRQ \Rightarrow$

- $P \models g(Q)$ and
- $\forall\ P \xrightarrow{a} P'$, $\exists\ Q \xrightarrow{a} Q'$ such that $P'RQ'$.

Definition 4.
Define P and Q to be forward g simulation equivalent, written as $P \sim_g Q$, iff there are forward g simulations R_1 and R_2 with PR_1Q and QR_2P.

As a first step to proving that a simulation like equivalence is a modal equivalence it turns out to be very useful to prove that it is a forward g-equivalence. Consider the formula $\mathsf{f}_g \in \mathbb{K}^{Par}$ defined by the abbreviated equations

- $\mathsf{f}_g(P) = \Box(\bigvee\{a \wedge \mathsf{f}_g(P') \mid P \overset{a}{\longmapsto} P'\}) \wedge g(P)$
- $\mathsf{f}_g(0) = \Box\,\mathsf{F}$

Lemma 5.
Let $g \in \mathbb{K}^{Par}$ and P, Q be process terms. There is a forward g simulation R such that PRQ, if and only if $P \models \mathsf{f}_g(Q)$.

Proof
For the proof we freely regard P and Q as finite graphs, with start nodes λ_P and λ_Q. For a graph G let $N(G)$ denote the node set of G, and for u, $v \in N(G)$ let $u \xrightarrow{a} v \in G$ denote there is an a labeled edge from u to v. Let $E(u) = \{u \xrightarrow{a} v \mid u \xrightarrow{a} v \in G\}$.

For a node u of G we write (G, u) to denote the subgraph of G which descends from u. For a predicate $\mathsf{f} \in F$ we will write $\mathsf{f}(G, u)$ rather than the unwieldy $\mathsf{f}((G, u))$.

Let R be a forward g simulation such that PRQ. Thus there is a map $\phi : N(P) \longrightarrow N(Q)$ such that $\phi(\lambda_P) = \lambda_Q$, and

- $\forall\ u \in N(P)$, $(P, u) \models g(Q, \phi(u))$, and
- $u \xrightarrow{a} v \in P \Rightarrow \phi(u) \xrightarrow{a} \phi(v) \in Q$

Let at(v) be a meta-logic predicate, where $(P, u) \models$ at (v) iff $u = v$. Clearly for any $u \in N(P)$,

$$(P, u) \models \text{at }(u) \wedge g(Q, \phi(u)) \wedge \Box(\bigvee\{a \wedge \text{at }(v) \mid u \xrightarrow{a} v \in P\})$$

Now define, for u a node of $N(P)$, the formula

$$\mathsf{f}_\phi(u) = \text{at }(u) \wedge g(Q, \phi(u)) \wedge \Box(\bigvee\{a \wedge \mathsf{f}_\phi(v) \mid u \xrightarrow{a} v \in P\})$$

Therefore, from the previous remark, we have

$$(P, u) \models \mathsf{f}_\phi(u)$$

It is not to hard to show that $(P, u) \models \mathsf{f}_\phi(u) \longrightarrow \mathsf{f}_g(Q, \phi(u))$, hence

$$(P, u) \models \mathsf{f}(Q, \phi(u))$$

Which completes the proof of the first part of the lemma.

Suppose that $P \models \mathsf{f}_g(Q)$. Define a relation δ over processes by $P\delta Q$ iff $P \models \mathsf{f}_g(Q)$. If δ is a forward g simulation the proof is complete.

Trivially $P\delta Q \Rightarrow P \models g(Q)$. Suppose that $P \xrightarrow{a} P'$. Then

$$a \cdot P' \models \Box(\bigvee\{b \wedge \mathsf{f}_g(Q,v) \mid \lambda_Q \xrightarrow{b} v \in Q\})$$

Thus there must be some b and some v such that $a = b$ and $P' \models \mathsf{f}_g(Q,v)$. That is putting $Q' = (Q,v)$, $P' \models \mathsf{f}_g(Q')$. This proves that δ is a forward g simulation which completes the proof of the lemma. $\Box$

There is a dual notion to forward g simulation, which is backward h simulation. Let $h \in \mathbb{K}^{Par}$ and define a relation R of processes to be a backward h simulation iff $PRQ \Rightarrow$

- $Q \models h(P)$ and
- $\forall\ P \xrightarrow{a} P'$, $\exists\ Q \xrightarrow{a} Q'$ such that $P'RQ'$.

Definition 6.
Define P and Q to be backward h simulation equivalent, written as $P \sim^h Q$, iff there are backward h simulations R_1 and R_2 with PR_1Q and QR_2P.

Not surprisingly it turns out to be useful, when proving a simulation like equivalence is a modal equivalence, to prove that it is a backward h simulation equivalence. In fact if one can prove that an equivalence is a forward g and backward h equivalence then it nearly follows the equivalence is a modal equivalence.

Define the formula f'_h by the abbreviated equations

- $\mathsf{f}'_h(P) = (\bigwedge\{\Diamond(a \wedge \mathsf{f}'_h(P')) \mid P \stackrel{a}{\longmapsto} P'\}) \wedge h(P)$
- $\mathsf{f}'_h(0) = \mathsf{T}$

The following lemma is the dual to lemma 5

Lemma 7.
Let $h \in \mathbb{K}^{Par}$ and P, Q be process terms. There is a backward h simulation R such that PRQ, if and only if $Q \models \mathsf{f}'_h(P)$.

Proof
The proof of this lemma is analogous to that of lemma 5. The notation introduced for that proof will be used here also.

Suppose that R is a backward h simulation and that PRQ. Then there is a map $\phi : N(P) \longrightarrow N(Q)$ such that $\phi(\lambda_P) = \lambda_Q$, and

- $\forall\ u \in N(P)$, $(Q, \phi(u)) \models h(P,u)$, and
- $u \xrightarrow{a} v \in P \Rightarrow \phi(u) \xrightarrow{a} \phi(v) \in Q$

Therefore

$$(Q, \phi(u)) \models \text{at } (\phi(u)) \wedge \bigwedge\{\Diamond(a \wedge \text{at } (\phi(v)) \wedge h(P,v)) \mid u \xrightarrow{a} v \in P\}$$

Now define a formula $\mathsf{f}'_\phi(u)$ when $u \in N(P)$ by

$$\mathsf{f}'_\phi(u) = \text{at } (\phi(u)) \wedge h(P,u) \wedge \bigwedge\{\Diamond(a \wedge \mathsf{f}'_\phi(v)) \mid u \xrightarrow{a} v \in P\}$$

Therefore, from the preceding remark

$$(Q, \phi(u)) \models \mathsf{f}'_\phi(u)$$

It can also be proved that $(Q, \phi(u)) \models \mathsf{f}'_\phi(u) \longrightarrow \mathsf{f}'_h(P, u)$, hence

$$(Q, \lambda_Q) \models \mathsf{f}'_h(P, \lambda_P)$$

This completes the first part of the proof.

Suppose that $Q \models \mathsf{f}'_h(P)$. Define a relation δ by $P\delta Q$ iff $Q \models \mathsf{f}'_h(P)$. Trivially if $P\delta Q$ then $Q \models h(P)$. Suppose that $P \xrightarrow{a} P'$ then

$$Q \models \Diamond(a \wedge \mathsf{f}'_h(P'))$$

This can only be true if for some $\lambda_Q \xrightarrow{a} v \in Q$, $(Q, v) \models \mathsf{f}'_h(P')$. That is, putting $Q' = (Q, v)$, $Q' \models \mathsf{f}'_h(P')$, so that δ is a backward h simulation. This completes the proof of the lemma.

□

Proposition 8.
Let g, $h \in \mathbb{K}^{Par}$. Suppose that g and h have the property that a relation R is a forward g simulation iff R is a backward h simulation. Therefore $\sim_g$ and $\sim^h$ are the same relation, let $\sim_\sigma$ denote this relation. Define $\mathsf{f}_\sigma(P) = \mathsf{f}_g(P) \wedge \mathsf{f}'_h(P)$. Then

$$P \sim_\sigma Q \Leftrightarrow P \models \mathsf{f}_\sigma(Q)$$

Proof
If $P \sim_\sigma Q$ then trivially $P \models \mathsf{f}_g(Q)$ by lemma 5, and $P \models \mathsf{f}'_h(Q)$ by lemma 7. Therefore $P \models \mathsf{f}_\sigma(Q)$.

Equally if $P \models \mathsf{f}_g(Q)$ then there is a forward simulation R such that PRQ by lemma 5. Similarly if $P \models \mathsf{f}'_h(Q)$ then there is a backward simulation S such that QSP. By the hypothesis S is also a forward simulation, hence $P \sim_\sigma Q$. □

During the rest of this section each of simulation, complete simulation, ready simulation, 2-nested simulation, weak bisimulation and congruence will be proved to be modal equivalences by application of proposition 8.

3.1 Simulation Equivalence

Simulation equivalence has been around in the literature for some time, see for example [20]. A relation R over processes is a *simulation* if

$$PRQ \Rightarrow \forall P \xrightarrow{a} P' \, \exists Q \xrightarrow{a} Q' : P'RQ'$$

Define $P \sim_S Q$ iff there are simulations R_1 and R_2 such that PR_1Q and QR_2P.

Take $g(t) = \mathsf{T}$ and $h(t) = \mathsf{T}$. Quite obviously a relation R is a forward g simulation iff it is a backward h simulation iff it is a simulation. Thus $\sim_S$ is a modal equivalence with characteristic formula f_S defined by $\mathsf{f}_S(t) = \mathsf{f}(t) \wedge \mathsf{f}'(t)$ where:

- $\mathsf{f}(P) = \Box(\bigvee\{a \wedge \mathsf{f}(P') \mid P \overset{a}{\longmapsto} P'\})$
- $\mathsf{f}(0) = \Box\,\mathsf{F}$

- $\mathsf{f}'(P) = \bigwedge\{\Diamond(a \wedge \mathsf{f}'(P')) \mid P \overset{a}{\longmapsto} P'\}$
- $\mathsf{f}'(0) = \mathsf{T}$

The lemmas 5, 7 and proposition 8 for this case prove that:

$$\text{There is a simulation } R \text{ s.t. } PRQ \text{ iff } P \models \mathsf{f}(Q)$$

$$\text{There is a simulation } R \text{ s.t. } QRP \text{ iff } P \models \mathsf{f}'(Q)$$

and finally $P \sim_S Q$ iff $P \models \mathsf{f}_S(Q)$.

For example consider $P = a \cdot (b \cdot 0 + c \cdot 0)$ and $Q = a \cdot (b \cdot 0 + c \cdot 0) + a \cdot b \cdot 0$. Then

$$\begin{aligned}
\mathsf{f}(P) &= \Box(a \wedge \Box(b \wedge \Box\,\mathsf{F} \vee c \wedge \Box\,\mathsf{F})) \\
\mathsf{f}'(P) &= \Diamond(a \wedge \Diamond\, b \wedge \Diamond\, c) \\
\mathsf{f}(Q) &= \Box(a \wedge \Box(b \wedge \Box\,\mathsf{F} \vee c \wedge \Box\,\mathsf{F}) \vee a \wedge (\Box\, b \wedge \Box\,\mathsf{F})) \\
\mathsf{f}'(Q) &= \Diamond(a \wedge \Diamond\, b \wedge \Diamond\, c) \wedge \Diamond(a \wedge \Diamond\, b)
\end{aligned}$$

It is indeed the case that $P \sim_S Q$ and it is straightforward to check the processes satisfy the relevant formulae.

3.2 Ready Simulation Equivalence

Ready simulation also goes by the name of 2/3-simulation, and was originally defined as GSOS trace congruence in [2]. For a process P let $I(P)$ be the complete set of initial transitions that can occur, that is

$$I(P) = \{a \mid \exists\, P' : \ P \overset{a}{\longrightarrow} P'\}$$

A relation R over processes is a *ready simulation* if

$$PRQ \Rightarrow I(P) = I(Q) \text{and } \forall P \overset{a}{\longrightarrow} P' \ \exists Q \overset{a}{\longrightarrow} Q' : P'RQ'$$

Write $P \sim_{RS} Q$ iff there are ready simulations R_1 and R_2 such that PR_1Q and QR_2P.

In this case take

$$\begin{aligned}
g(P) &= \bigwedge\{\Diamond\, a \mid P \overset{a}{\longrightarrow} P'\} \\
h(P) &= \Box(\bigvee\{a \mid P \overset{a}{\longrightarrow} P'\})
\end{aligned}$$

If a relation R of processes is a forward g simulation, then by definition if PRQ, $P \models g(Q)$. It is fairly easy to see that this can only be so if $I(Q) \subseteq I(P)$. Similarly if R is a backward h simulation then PRQ implies that $Q \models h(P)$. If this is true then $I(Q) \subseteq I(P)$.

Recall that

$$\mathsf{f}_g(P) = \Box(\bigvee\{a \wedge \mathsf{f}_g(P') \mid P \overset{a}{\longmapsto} P'\}) \wedge g(P)$$

Clearly for any t, $\mathsf{f}_g(t) \longrightarrow h(t)$. By lemma 5, if R is a forward g simulation then $PRQ \Rightarrow P \models \mathsf{f}_g(Q)$. Thus $P \models h(Q)$, so that a forward g simulation is also a forward $g \wedge h$ simulation.

It is also true that for any t, $\mathsf{f}'_h(t) \longrightarrow g(t)$. Thus a backward h simulation is also a backward $g \wedge h$ simulation. Moreover if $P \models g(Q) \wedge h(Q)$ then $I(P) = I(Q)$. Therefore a relation R is a ready simulation, iff R is a forward $g \wedge h$ simulation, iff R is a backward $g \wedge h$ simulation.

We have, as mentioned, $\mathsf{f}_g(t) \longrightarrow h(t)$ and $\mathsf{f}'_h(t) \longrightarrow g(t)$. Therefore $\mathsf{f}_{g\wedge h}$ is equivalent to f_g, and $\mathsf{f}'_{g\wedge h}$ is equivalent to f'_h. Therefore we can define $\mathsf{f}_{RS}(P) = \mathsf{f}_g(P) \wedge \mathsf{f}'_h(P)$. From proposition 8, $P \sim_{RS} Q$ iff $P \models \mathsf{f}_{RS}(Q)$. Ready simulation is therefore a modal simulation with characteristic formula f_{RS}.

For example consider $P = a \cdot (b \cdot 0 + c \cdot 0)$ and $Q = a \cdot (b \cdot 0 + c \cdot 0) + a \cdot b \cdot 0$. Then

$$
\begin{aligned}
\mathsf{f}_g(P) &= \Box(a \wedge \Box(b \wedge \Box\,\mathsf{F} \vee c \wedge \Box\,\mathsf{F}) \wedge \Diamond\, b \wedge \Diamond\, c) \wedge \Diamond\, a \\
\mathsf{f}'_h(P) &= \Diamond(a \wedge \Diamond\, b \wedge \Diamond\, c \wedge \Box(b \vee c)) \wedge \Box\, a \\
\mathsf{f}_g(Q) &= \Box(a \wedge \Box(b \wedge \Box\,\mathsf{F} \vee c \wedge \Box\,\mathsf{F}) \\
&\qquad \wedge \Diamond\, b \wedge \Diamond\, c \vee a \wedge (\Box\, b \wedge \Box\,\mathsf{F}) \wedge \Diamond\, b) \wedge \Diamond\, a \\
\mathsf{f}'_h(Q) &= \Diamond(a \wedge \Diamond\, b \wedge \Diamond\, c \wedge \Box(b \vee c)) \wedge \Diamond(a \wedge \Diamond\, b \wedge \Box\, b) \wedge \Box\, a
\end{aligned}
$$

In this example $Q \models \mathsf{f}'_h(P)$ but $Q \not\models \mathsf{f}_g(P)$. Whereas $P \models \mathsf{f}_g(Q)$ but $P \not\models \mathsf{f}'_h(Q)$. So that $P \not\sim_{RS} Q$.

3.3 Complete Simulation

Complete simulation, to be found in [6], is in some ways a dual to ready simulation, the subject of the last section. A relation R over processes is a *complete simulation* if PRQ implies

- $I(P) = \emptyset \Leftrightarrow I(Q) = \emptyset$ and
- $\forall P \xrightarrow{a} P' \; \exists Q \xrightarrow{a} Q' : P'RQ'$

Write $P \sim_{CS} Q$ iff there are complete simulations R_1 and R_2 such that PR_1Q and QR_2P.

Complete simulation does not quite fit into the framework for generating a characteristic formula, as used for simulation and ready simulation. Define two functions $\mathsf{f}, \mathsf{f}' \in F$ with the abbreviated equations:

- $\mathsf{f}(P) = \Box(\bigvee\{a \wedge \mathsf{f}(P') \mid P \overset{a}{\longmapsto} P'\})$
- $\mathsf{f}(0) = \Box\,\mathsf{F}$
- $\mathsf{f}'(P) = \bigwedge\{\Diamond(a \wedge \mathsf{f}'(P')) \mid P \overset{a}{\longmapsto} P'\}$
- $\mathsf{f}'(0) = \Box\,\mathsf{F}$

These equations are almost identical to those which define f_S. The only difference is that for complete simulation $\mathsf{f}'(0) = \Box\,\mathsf{F}$, whereas in the case of simulation $\mathsf{f}'(0) = \mathsf{T}$. Define $\mathsf{f}_{CS} = \mathsf{f} \wedge \mathsf{f}'$. This is the characteristic equation for complete simulation, that is $P \sim_{CS} Q$ iff $P \models \mathsf{f}_{CS}(Q)$.

One can modify the definition of forward and backward simulations to invent the idea of a complete forward and complete backward simulation. For example, a complete forward g simulation is a relation R such that if PRQ then

- $I(P) = \emptyset \Leftrightarrow I(Q) = \emptyset$ and
- $P \models g(Q)$ and

- $\forall P \xrightarrow{a} P' \; \exists Q \xrightarrow{a} Q' : P'RQ'$

One can modify lemmas 5 and 7 so that they refer to complete forward g simulations, and complete backward h simulations. The result requires slightly different formulae for f_g and f'_h. The new formula f_g is identical to the original f_g. The new f'_h differs from the old version only in that $\mathsf{f}'_h(0)$ is now defined to be $\Box\,\mathsf{F}$ rather than T.

There is a complete version of proposition 8. The proof that f_{CS} is the characteristic formula for $\sim_{CS}$ is analogous to the proof that f_S is the characteristic formula for $\sim_S$.

3.4 2-nested simulation equivalence

The 2-nested simulation equivalence appeared in a paper by Groote and Vaandrager [9] in 1988. A relation R over processes is a *2-nested simulation* if PRQ implies

- $P \sim_S Q$ and
- $\forall P \xrightarrow{a} P' \; \exists Q \xrightarrow{a} Q' : P'RQ'$

Define$P \sim_{2S} Q$ iff there are 2-nested simulations R_1 and R_2 such that PR_1Q and QR_2P.

Let f and f' be the functions defined in 3.1. Recall that $\mathsf{f}_S = \mathsf{f} \wedge \mathsf{f}'$. Take $g = \mathsf{f}'$ and $h = \mathsf{f}$. Let f_g and f'_h be the formulae appearing in lemmas 5 and 7. Define $\mathsf{f}_{2S} = \mathsf{f}_g \wedge \mathsf{f}'_h$. This is the characteristic formula for $\sim_{2S}$.

It is easy to see that $\mathsf{f}_g \longrightarrow \mathsf{f}$, and that $\mathsf{f}'_h \longrightarrow \mathsf{f}'$. So if a relation R is a forward g simulation and PRQ, it follows that $P \models \mathsf{f}_g(Q)$ by lemma 5. Thus $P \models \mathsf{f}'(Q)$. Since $\mathsf{f}_g \longrightarrow \mathsf{f}$, it also follows that $P \models \mathsf{f}(Q)$. Hence $P \models \mathsf{f}_S(Q)$, and $P \sim_S Q$. Similarly $\mathsf{f}'_h \longrightarrow \mathsf{f}'$, from which it follows that if $Q \models \mathsf{f}'_h(P)$ then $Q \models \mathsf{f}_S(P)$, and$Q \sim_S P$. Hence $\mathsf{f}_{g\wedge h} \leftrightarrow \mathsf{f}_g$ and $\mathsf{f}'_{g\wedge h} \leftrightarrow \mathsf{f}'_h$.

It is clear that a relation R is a 2-nested simulation iff R is a forward $g \wedge h$ simulation, iff R is a backward $g \wedge h$ simulation. The formula $\mathsf{f}_{g\wedge h}$ is equivalent to f_g, and $\mathsf{f}'_{g\wedge h}$ is equivalent to f'_h. Therefore f_{2S} is the characteristic formula for $\sim_{2S}$. That is, $P \sim_{2S} Q$ iff $P \models \mathsf{f}_{2S}(Q)$.

Consider for example $P = a \cdot b \cdot c \cdot 0 + a \cdot (b \cdot c \cdot 0 + b \cdot d \cdot 0)$, and $Q = a \cdot (b \cdot c \cdot 0 + b \cdot d \cdot 0)$. In this case $\mathsf{f}_g(Q)$ will be

$$\Box(a \wedge \Box(b \wedge \Diamond\, c \wedge \Box(c \wedge \Box\,\mathsf{F}) \vee \\ b \wedge \Diamond\, d \wedge \Box(d \wedge \Box\,\mathsf{F}))) \wedge \Diamond(a \wedge (\Diamond\, b \wedge \Diamond\, c) \wedge (\Diamond\, b \wedge \Diamond\, d))$$

One can prove that $P \not\models \mathsf{f}_g(Q)$. Suppose not, if $P \models \mathsf{f}_g(Q)$, then $P \models \mathsf{f}'(Q)$. However

$$\mathsf{f}'(Q) = \Diamond(a \wedge (\Diamond\, b \wedge \Diamond\, c) \wedge (\Diamond\, b \wedge \Diamond\, d)).$$

If $P \models \mathsf{f}'(Q)$, then $(a \cdot b \cdot c \cdot 0) \models \mathsf{f}'(Q)$. This is not true, $(a \cdot b \cdot c \cdot 0) \not\models \mathsf{f}'(Q)$, hence $P \not\models \mathsf{f}_g(Q)$ and $P \not\sim_{2S} Q$.

This is an interesting example as it shows how small fragments of f_{2S} are all that are required sometimes to tell apart two processes. If two processes are distinguished by a particular modal equivalence, it is possible this distinction can be made by considering a logical consequence of the characteristic formula, and using that to tell them apart. This provides a useful heuristic method of determining when two processes are different. Choose simpler formulae which are implied by the characteristic formula, and try those.

3.5 Strong bisimulation equivalence

Strong bisimulation [16] is at the heart of all the observational equivalences which appear in the literature. Any equivalence which is classified in van Glabbeek's paper [7] contains strong bisimulation.

The fact that strong bisimulation is a modal equivalence, and so has a characteristic formula has really been known since [16]. Though Milner does not explicitly state the result in that form, and does not use exactly the same modal logic as $\mu\mathbb{K}$. Walker describes how to formulate a characteristic formula for $\sim$ in [25] using Hennessey Milner logic. Walker does not make the observation that this characteristic formula exists as a formula in a predicate modal logic such as $\mathbb{K}^{Par}$.

Recall that a relation R is a strong bisimulation equivalence if PRQ implies

- $\forall\ P \xrightarrow{a} P',\ \exists\ Q \xrightarrow{a} Q' : P'RQ'$ and
- $\forall\ Q \xrightarrow{a} Q',\ \exists\ P \xrightarrow{a} P' : Q'RP'$

Process P and Q are strong bisimulation equivalent if there is a strong bisimulation R such that PRQ. This equivalence is denoted $\sim$. Define

$$\mathsf{f}_{\sim}(P) = \Box(\bigvee\{a \wedge \mathsf{f}_{\sim}(P') \mid P \stackrel{a}{\longmapsto} P'\}) \wedge \bigwedge\{\Diamond(a \wedge \mathsf{f}_{\sim}(P')) \mid P \stackrel{a}{\longmapsto} P'\}$$

and $\mathsf{f}_{\sim}(0) = \Box\,\mathsf{F}$. This formula is the characteristic formula for $\sim$.

Recall that we identify image finite processes with their graph representation. So that for a process P we can refer to the nodes $N(P)$ of the graph representation of P. In this section we use the notation introduced in the proofs of lemmas 5 and 7.

There is a strong bisimulation equivalence R with PRQ iff there are functions $f_1 : N(P) \longrightarrow N(Q)$ and $f_2 : N(Q) \longrightarrow N(P)$ such that

- for $u,\, v \in N(P)$, $u \xrightarrow{a} v \in P$ implies $f_1(u) \xrightarrow{a} f_1(v) \in Q$
- for $u \in N(P)$, $v \in N(Q)$, $f_1(u) \xrightarrow{a} v \in Q$ implies $u \xrightarrow{a} f_2(v) \in P$
- for $u,\, v \in N(Q)$, $u \xrightarrow{a} v \in Q$ implies $f_2(u) \xrightarrow{a} f_2(v) \in P$
- for $u \in N(Q)$, $v \in N(P)$, $f_2(u) \xrightarrow{a} v \in P$ implies $u \xrightarrow{a} f_1(v) \in Q$

Also $f_1(\lambda_P) = \lambda_Q$ and $f_2(\lambda_Q) = \lambda_P$. From this one can see that

$$(P, u) \models \left(\begin{array}{c} \text{at } u \wedge \Box(\bigvee\{a \wedge \text{at } f_2(v) \mid f_1(u) \xrightarrow{a} v \in Q\}) \\ \wedge \\ \bigwedge\{\Diamond(a \wedge \text{at } f_2(v) \mid f_1(u) \xrightarrow{a} v \in Q\} \end{array} \right) \tag{$\dagger$}$$

Now define a formula $\mathsf{f}(u)$ for $u \in N(P)$ by

$$\begin{aligned} \mathsf{f}(u) = \Box(\bigvee\{a \wedge \mathsf{f}(f_2(v)) \mid f_1(u) \xrightarrow{a} v \in Q\}) \wedge \\ \bigwedge\{\Diamond(a \wedge \mathsf{f}(f_2(v)) \mid f_1(u) \xrightarrow{a} v \in Q\} \end{aligned}$$

It follows from ($\dagger$) that $(P, u) \models \mathsf{f}(u)$, it is also true that $\mathsf{f}(u) \longrightarrow \mathsf{f}_{\sim}(Q, f_1(u))$. Therefore $(P, u) \models \mathsf{f}_{\sim}(Q)$. Hence, $P \sim Q \Rightarrow P \models \mathsf{f}_{\sim}(Q)$.

Suppose then that $P \models \mathsf{f}_\sim(Q)$. It is necessary to prove that $P \sim Q$. Define formulae g, $h \in \mathbb{K}^{Par}$ by

$$g = \mathsf{f}'_h,\ h = \mathsf{f}_g$$

Thus written out more fully

$$\begin{aligned}\mathsf{f}_g(P) &= \Box(\bigvee\{a \wedge \mathsf{f}_g(P') \mid P \overset{a}{\longmapsto} P'\}) \wedge \mathsf{f}'_h(P)\\ \mathsf{f}'_h(P) &= (\bigwedge\{\Diamond(a \wedge \mathsf{f}'_h(P')) \mid P \overset{a}{\longmapsto} P'\}) \wedge \mathsf{f}_g(P)\end{aligned}$$

These two equations are mutually recursive. Careful observation shows that $\mathsf{f}_\sim = \mathsf{f}_g = \mathsf{f}'_h$.

Define $P\delta Q$ if $P \models \mathsf{f}_\sim(Q)$. We can prove relation δ is a strong bisimulation equivalence. If $P\delta Q$ then δ is a forward g simulation, this was proved as part of the proof of lemma 5. Thus $\forall\ P \xrightarrow{a} P'$, $\exists\ Q \xrightarrow{a} Q' : P'\delta Q'$. Of course from the definition of f_g, if $P\delta Q$, then $P \models \mathsf{f}'_h(Q)$. And from the proof of lemma 7 it follows that δ is also a backward h simulation, therefore $\forall$ $Q \xrightarrow{a} Q'$, $\exists\ P \xrightarrow{a} P' : Q'\delta P'$. This completes the proof. Thus $P \sim Q$ iff $P \models \mathsf{f}_\sim(Q)$.

The final part of the proof expresses $\mathsf{f}_\sim$ as a g forward simulation and as a h backward simulation where g and h are both $\mathsf{f}_\sim$!

3.6 Weak Bisimulation equivalence

This section and the next deal with equivalences which incorporate properties of the silent action τ. Write $P \longrightarrow^* Q$ to denote that there is a sequence $P_i \xrightarrow{\tau} P_{i+1}$, where $P_0 = P$ and $P_n = Q$ for some $n \geq 0$. Write $P \overset{a}{\leadsto} Q$ to denote that $P \longrightarrow^* P' \xrightarrow{a} Q' \longrightarrow^* Q$. If $a = \tau$, let a^* be ϵ, the empty transition, otherwise $a^* = a$. The empty transition has the property that if $P \xrightarrow{\epsilon} P'$, then $P = P'$. It is introduced simply to make the notation work better. Notice that $P \overset{\tau^*}{\leadsto} P$ with this notation. Historically, $\leadsto$ is written as $\Rightarrow$, here we have avoided this notation because of the obvious possible confusion with logical implication, also denoted by $\Rightarrow$.

Weak bisimulation is a modal equivalence, there is a formula $\mathsf{f}_\approx \in \mathbb{K}^{Par}$ such that $P \approx Q$ iff $P \models \mathsf{f}_\approx(Q)$.

A relation R is a weak bisimulation equivalence if when PRQ,

- $\forall P \xrightarrow{a} P'\ \exists Q \overset{a^*}{\leadsto} Q' : P'RQ'$ and
- $\forall Q \xrightarrow{a} Q'\ \exists P \overset{a^*}{\leadsto} P' : Q'RP'$.

Processes P and Q are weak bisimulation equivalent, written as $P \approx Q$, if there is a weak bisimulation R, such that PRQ.

For a graph G and nodes u, v of G, give the statement $u \overset{a}{\leadsto} v$ the analogous meaning to $P \overset{a}{\leadsto} Q$. That is, there is a path from u to v where exactly one edge in the path is labeled by a, and all the other edges in the path are labeled by τ. There is a weak bisimulation equivalence R with PRQ iff there are functions $f_1 : N(P) \longrightarrow N(Q)$ and $f_2 : N(Q) \longrightarrow N(P)$ such that

- for u, $v \in N(P)$, $u \xrightarrow{a} v \in P$ implies $f_1(u) \overset{a^*}{\leadsto} f_1(v) \in Q$
- for $u \in N(P)$, $v \in N(Q)$, $f_1(u) \xrightarrow{a} v \in Q$ implies $u \overset{a^*}{\leadsto} f_2(v) \in P$
- for u, $v \in N(Q)$, $u \xrightarrow{a} v \in Q$ implies $f_2(u) \overset{a^*}{\leadsto} f_2(v) \in P$

- for $u \in N(Q)$, $v \in N(P)$, $f_2(u) \xrightarrow{a} v \in P$ implies $u \overset{a^*}{\rightsquigarrow} f_1(v) \in Q$

Also $f_1(\lambda_P) = \lambda_Q$ and $f_2(\lambda_Q) = \lambda_P$.

Next we define some notation to make the definition of the characteristic formula $\mathsf{f}_{\approx}$ more digestible. Define

$$\begin{aligned}
\Diamond_\tau f &= \mu X.\,(f \vee \tau \wedge \Diamond X) \\
\Box_\tau f &= \mu X.\,(f \vee \tau \wedge \Box X) \\
\Diamond_a^* f &= \ ((a \leftrightarrow \tau) \longrightarrow \Diamond_\tau f) \\
&\ \wedge \\
&\ (\neg(a \leftrightarrow \tau) \longrightarrow \Diamond_\tau (a \wedge f)) \\
\Box_a^* f &= \ ((a \leftrightarrow \tau) \longrightarrow \Box_\tau f) \\
&\ \wedge \\
&\ (\neg(a \leftrightarrow \tau) \longrightarrow \Box_\tau (a \wedge f))
\end{aligned}$$

The formula $\Diamond_\tau f$ is true if there is some sequence of τ actions after which f becomes true. The formula $\Box_\tau f$ is true at node u of a model, if every path out of u is a chain of τ actions followed by f. Formula $\Diamond_a^* f$ is true if, either $a = \tau$ and $\Diamond_\tau f$ is true, or $a \neq \tau$ and $\Diamond_\tau (a \wedge f)$ is true. Formula $\Box_a^* f$ works in an analogous fashion.

Define $\mathsf{f}_{\approx}$ by the abbreviated equations:

$$\mathsf{f}_{\approx}(P) = \left(\begin{array}{l} \Box(\bigvee\{\Box_a^* \mathsf{f}_{\approx}(P') \mid P \overset{a}{\longmapsto} P'\}) \\ \wedge \\ \bigwedge\{\Diamond\, \Diamond_a^* \mathsf{f}_{\approx}(P') \mid P \overset{a}{\longmapsto} P'\} \end{array} \right)$$

and $\mathsf{f}_{\approx}(0) = \Box_\tau \mathsf{F}$. At first sight this seems to be a much larger formula than previous characteristic formulae. However it is linear in the length of $\mathsf{f}_{\sim}$. If P has structural depth n, and maximum fan out m, then $\mathsf{f}_{\approx}$ is linear in nm.

The proof that $\mathsf{f}_{\approx}$ characterises $\approx$ is analogous to the proof that $\mathsf{f}_{\sim}$ characterises $\sim$.

Suppose that R is a weak bisimulation equivalence and that PRQ. Let f_1 and f_2 be maps as defined above. One can see that

$$(P, u) \models \left(\begin{array}{l} \mathrm{at}\ u \wedge \Box(\bigvee\{\Box_a^* \,\mathrm{at}\ f_2(v) \mid f_1(u) \xrightarrow{a} v \in Q\}) \\ \wedge \\ \bigwedge\{\Diamond\, \Diamond_a^* \,\mathrm{at}\ f_2(v) \mid f_1(u) \xrightarrow{a} v \in Q\} \end{array} \right) \tag{‡}$$

Now define a formula $\mathsf{f}(u)$ for $u \in N(P)$ by

$$\mathsf{f}(u) = \Box(\bigvee\{\Box_a^* \mathsf{f}(f_2(v)) \mid f_1(u) \xrightarrow{a} v \in Q\}) \wedge \\ \bigwedge\{\Diamond\, \Diamond_a^* \,\mathrm{at}\ f_2(v) \mid f_1(u) \xrightarrow{a} v \in Q\}$$

It follows from (‡) that $(P, u) \models \mathsf{f}(u)$, it is also true that $\mathsf{f}(u) \longrightarrow \mathsf{f}_{\approx}(Q, f_1(u))$. Therefore $(P, u) \models \mathsf{f}_{\approx}(Q)$. Hence, $P \approx Q \Rightarrow P \models \mathsf{f}_{\approx}(Q)$.

Suppose now that $P \models \mathsf{f}_{\approx}(Q)$. Observe that this is equivalent to constructing two maps $g_1 : N(P) \longrightarrow N(Q)$ and $g_2 : N(Q) \longrightarrow N(P)$ with the following properties:

- for u, $v \in N(P)$, $u \overset{a^*}{\rightsquigarrow} v \in P$ implies $g_1(u) \overset{a^*}{\rightsquigarrow} g_1(v) \in Q$

- for $u \in N(P)$, $v \in N(Q)$, $g_1(u) \xrightarrow{a} v \in Q$ implies $u \overset{a^*}{\rightsquigarrow} g_2(v) \in P$
- for u, $v \in N(Q)$, $u \xrightarrow{a} v \in Q$ implies $g_2(u) \overset{a^*}{\rightsquigarrow} g_2(v) \in P$
- for $u \in N(Q)$, $v \in N(P)$, $g_2(u) \overset{a^*}{\rightsquigarrow} v \in P$ implies $u \overset{a^*}{\rightsquigarrow} g_1(v) \in Q$

Also $g_1(\lambda_P) = \lambda_Q$ and $g_2(\lambda_Q) = \lambda_P$. This is in fact more than is required to construct a weak bisimulation equivalence between P and Q. If we put $f_1 = g_1$ and $f_2 = g_2$ then the pair f_1, f_2 is a weak bisimulation equivalence between P and Q, hence $P \approx Q$.

3.7 Congruence

Congruence [16] is a fascinating equivalence relation. It can be characterised by the τ-laws for finite processes. These converted into modal logic axioms [19] provide an elegant way of constructing a denotational semantics for congruence.

Process P is congruent to Q, written as $P =_\tau Q$, if

- for every $P \xrightarrow{a} P'$ there is a $Q \xrightarrow{a} Q'$ such that $P' \approx Q'$, and
- for every $Q \xrightarrow{a} Q'$ there is a $P \xrightarrow{a} P'$ such that $Q' \approx P'$.

This is very easy to encode in $\mathbb{K}^{Par}$ since we have already encoded $\approx$. Define f_τ by the abbreviated equations:

- $$\begin{aligned} \mathsf{f}_\tau(P) \quad = \quad & \Box(\bigvee\{a \wedge \mathsf{f}_\approx(P') \mid P \overset{a}{\longmapsto} P'\}) \\ & \wedge \\ & \bigwedge\{\Diamond(a \wedge \mathsf{f}_\approx(P')) \mid P \overset{a}{\longmapsto} P'\} \end{aligned}$$
- $\mathsf{f}_\tau(0) = \Box\,\mathsf{F}$.

For processes P and Q it is simple to prove that $P =_\tau Q$ if and only if $P \models \mathsf{f}_\tau(Q)$.

3.8 Template for Modal Simulation

The characteristic equations defined so far have fallen into a pattern. This is mostly because the equivalences could be defined as forward, and backward simulations. Suppose that g, $h \in \mathbb{K}^{Par}$ are defined by abbreviated equations (and so $g(\nu X.t)$ is guaranteed to expand as $\nu X.g(t)$, and $g(X)$ must be X, similarly for h). Let f_g and f'_h be defined by the abbreviated equations:

- $\mathsf{f}_g(P) = \Box(\bigvee\{a \wedge \mathsf{f}_g(P') \mid P \overset{a}{\longmapsto} P'\}) \wedge g(P)$
- $\mathsf{f}_g(0) = g(0) = \Box\,\mathsf{F}$
- $\mathsf{f}'_h(P) = (\bigwedge\{\Diamond(a \wedge \mathsf{f}'_h(P')) \mid P \overset{a}{\longmapsto} P'\}) \wedge h(P)$
- $\mathsf{f}'_h(0) = h(0) = \mathsf{T}$

Notice that this is virtually identical to the definitions from lemma 5, and lemma 7. The only difference is, the definitions here have the extra clauses, $g(0) = \Box\,\mathsf{F}$ and $h(0) = \mathsf{T}$.

Further suppose that g and h are connected by the implications:

- $\Box(\bigvee\{a \wedge h(P') \mid P \overset{a}{\longmapsto} P'\}) \wedge g(P) \longrightarrow h(P)$

- $(\bigwedge\{\Diamond(a \wedge g(P')) \mid P \overset{a}{\longmapsto} P'\}) \wedge h(P) \longrightarrow g(P)$

These are the template implications. Formula f_g and f'_h define the template which generates the infinite families promissed in the introduction. Each simulation in an infinite family will have a characteristic equation of the form $\mathsf{f}_g \wedge \mathsf{f}'_h$.

Proposition 9.
Let g and h be as defined above. Any forward g simulation is also a forward h simulation. Further, any backward h simulation is also a backward g simulation.

Write $P \sim_\sigma Q$ to denote that $P \models \mathsf{f}_g(Q) \wedge \mathsf{f}'_h(Q)$. Then $\sim_\sigma$ is a forward $g \wedge h$ simulation equivalence, and a backward $g \wedge h$ simulation equivalence.

Proof
A straightforward structural induction on P proves that $\mathsf{f}_g(P) \longrightarrow h(P)$. This induction works because, if $\mathsf{f}_g(P') \longrightarrow h(P')$ for each $P \overset{a}{\longmapsto} P'$, then

$$\mathsf{f}_g(P) \longrightarrow \Box(\bigvee\{a \wedge h(P') \mid P \overset{a}{\longmapsto} P'\}) \wedge g(P)$$

Therefore $\mathsf{f}_g(P') \longrightarrow h(P')$, using the implication above and the implication linking h and g. Similarly, $\mathsf{f}'_h(P) \longrightarrow g(P)$.

By definition this proves that any forward g simulation is also a forward h simulation, and, any backward h simulation is also a backward g simulation.

The last part of the proposition is now a direct consequence of proposition 8 because $\mathsf{f}_g \leftrightarrow \mathsf{f}_{g\wedge h}$, and $\mathsf{f}'_h \leftrightarrow \mathsf{f}'_{g\wedge h}$. This completes the proof. $\Box$

Let $\mathsf{f} \in \mathbb{K}^{Par}$ be a formula defined by $\mathsf{f} = \mathsf{f}_g \wedge \mathsf{f}'_h$, where g and h satisfy the template implications. Let $P \sim_{\sigma(\mathsf{f})} Q$ denote that $P \models \mathsf{f}(Q)$. Then $\sigma(\mathsf{f})$ is a sensible simulation equivalence in the following sense. Relation $P \sim_{\sigma(\mathsf{f})} Q$ is true iff there are forward g simulations R_1 and R_2 where PR_1Q and QR_2P; or equivalently there are backward h simulations R_1 and R_2 where PR_1Q and QR_2P.

There is also a template which defines parametric formulae which characterise weak simulation equivalences.

Definition 10.
Let $g,\ h \in \mathbb{K}^{Par}$. A relation R is a weak forward g simulation if PRQ implies

- $P \models g(Q)$ and
- $\forall\ P \overset{a}{\longrightarrow} P'$, $\exists\ Q \overset{a^*}{\rightsquigarrow} Q'$ such that $P'RQ'$.

A relation R is a weak backward h simulation if PRQ implies

- $Q \models h(P)$ and
- $\forall\ P \overset{a}{\longrightarrow} P'$, $\exists\ Q \overset{a^*}{\rightsquigarrow} Q'$ such that $P'RQ'$.

Define $P \approx_g Q$ iff there are weak forward g simulations R_1 and R_2 with PR_1Q and QR_2P. Similarly define $P \approx^h Q$ iff there are weak backward h simulations R_1 and R_2 with PR_1Q and QR_2P

Just as before a weak forward or backward simulation can be characterised by a formula of $\mathbb{K}^{Par}$. Define

- $w\mathsf{f}_g(P) = \Box(\bigvee\{\Box_a^* \, w\mathsf{f}_g(P') \mid P \overset{a}{\longmapsto} P'\}) \wedge g(P)$, $w\mathsf{f}_g(0) = \Box_\tau \, \mathsf{F}$
- $w\mathsf{f}'_h(P) = (\bigwedge\{\Diamond \, \Diamond_a^* \, w\mathsf{f}_h(P') \mid P \overset{a}{\longmapsto} P'\}) \wedge h(P)$, $w\mathsf{f}'_h(0) = \mathsf{T}$.

The formulae $\Box_a^*$ and $\Diamond_a^*$ are defined in section 3.6. The following lemma is proved in an analogous way to lemmas 5 and 7.

Lemma 11.
Let $\approx_g$ be a weak forward simulation equivalence defined by $g \in \mathbb{K}^{Par}$. Then there is a weak forward g simulation R, where PRQ if and only if $P \models w\mathsf{f}_g(Q)$.

Let $\approx^h$ be a weak backward simulation equivalence defined by $h \in \mathbb{K}^{Par}$. Then there is a weak backward h simulation R, where PRQ if and only if $Q \models w\mathsf{f}'_h(P)$.

Proposition 12.
Let g, $h \in \mathbb{K}^{Par}$. Suppose that g and h have the property that a relation R is a weak forward g simulation iff R is a weak backward h simulation. Therefore $\approx_g$ and $\approx^h$ are the same relation, let $\approx_\sigma$ denote this relation. Define $w\mathsf{f}_\sigma(P) = w\mathsf{f}_g(P) \wedge w\mathsf{f}'_h(P)$. Then

$$P \approx_\sigma Q \Leftrightarrow P \models w\mathsf{f}_\sigma(Q)$$

This proposition is proved analogously to proposition 8. This provides all the necessary machinery to define a weak template for formulae $w\mathsf{f}_g$ and $w\mathsf{f}'_h$.

Let h and g be formulae which are connected by the implications:

- $\Box(\bigvee\{\Box_a^* \, h(P') \mid P \overset{a}{\longmapsto} P'\}) \wedge g(P) \longrightarrow h(P)$
- $(\bigwedge\{\Diamond \, \Diamond_a^* \, g(P') \mid P \overset{a}{\longmapsto} P'\}) \wedge h(P) \longrightarrow g(P)$

Suppose also that $g(0) = \Box_\tau \, \mathsf{F}$ and $h(0) = \mathsf{T}$ These are the template implications. Formula $w\mathsf{f}_g$ and $w\mathsf{f}'_h$ define the template which generates the infinite family promised in the introduction. Each weak simulation in the family will have a characteristic equation of the form $w\mathsf{f}_g \wedge w\mathsf{f}'_h$.

Let $w\mathsf{f} \in \mathbb{K}^{Par}$ be a formula defined by $w\mathsf{f} = w\mathsf{f}_g \wedge w\mathsf{f}'_h$, where g and h satisfy the weak template implications. Let $P \approx_{\sigma(w\mathsf{f})} Q$ denote that $P \models w\mathsf{f}(Q)$. Then $\sigma(w\mathsf{f})$ is a 'sensible' weak simulation equivalence in the following sense. Relation $P \approx_{\sigma(w\mathsf{f})} Q$ is true iff there are weak forward g simulations R_1 and R_2 where PR_1Q and QR_2P; or equivalently there are weak backward h simulations R_1 and R_2 where PR_1Q and QR_2P.

4 Infinite Families

This section defines various infinite families of modal equivalences. Two of these families will converge to $\sim$, another will converge to $\approx$ and another will converge to $=_\tau$.

The first infinite family is based on generalising 2-nested simulation equivalence to n-nested simulation equivalence. Recall that f_S, (defined in section 3.1), is the characteristic formula for simulation equivalence. Formula $\mathsf{f}_S(t) = \mathsf{f}(t) \wedge \mathsf{f}'(t)$ where:

- $\mathsf{f}(P) = \Box(\bigvee\{a \wedge \mathsf{f}(P') \mid P \overset{a}{\longmapsto} P'\})$
- $\mathsf{f}(0) = \Box \, \mathsf{F}$
- $\mathsf{f}'(P) = \bigwedge\{\Diamond(a \wedge \mathsf{f}'(P')) \mid P \overset{a}{\longmapsto} P'\}$

- $\mathsf{f}'(0) = \mathsf{T}$

Define $\mathsf{f}_{(n+1)}$, and $\mathsf{f}'_{(n+1)}$ by the abbreviated equations

- $\mathsf{f}_{n+1}(P) = \Box(\bigvee\{a \wedge \mathsf{f}_{(n+1)}(P') \mid P \overset{a}{\longmapsto} P'\}) \wedge \mathsf{f}'_n(P)$
- $\mathsf{f}_{n+1}(0) = \Box\,\mathsf{F}$
- $\mathsf{f}'_{n+1}(P) = \bigwedge\{\Diamond(a \wedge \mathsf{f}'_{n+1}(P')) \mid P \overset{a}{\longmapsto} P'\} \wedge \mathsf{f}_n(P)$
- $\mathsf{f}'_{n+1}(0) = \mathsf{T}$

Define $\mathsf{f}_1 = \mathsf{f}$, and $\mathsf{f}'_1 = \mathsf{f}'$. Define $\mathsf{f}_{nS} = \mathsf{f}_n \wedge \mathsf{f}'_n$. Notice that $\mathsf{f}_{1S} = \mathsf{f}_S$, and of course f_{2S} has already been defined in section 3.4. All these formula are defined in accordance with the template of section 3.8.

Define the n-nested equivalence relation $\sim_{nS}$ by $P \sim_{nS} Q$ iff $P \models \mathsf{f}_{nS}(Q)$. From the definition and proposition 8, it follows that nS is a forward $\mathsf{f}_{(n-1)S}$ simulation equivalence and a backward $\mathsf{f}_{(n-1)S}$ simulation equivalence.

To spell this out in more detail. If there is an nS simulation R such that PRQ then there are $(n-1)S$ simulations R_1 and R_2 such that PR_1Q and QR_2P, also $\forall\ P \overset{a}{\longrightarrow} P'$, $\exists\ Q \overset{a}{\longrightarrow} Q'$ such that $P'RQ'$. Relation $P \sim_{nS} Q$ holds iff there are nS simulations R_1 and R_2 where PR_1Q and QR_2P.

Lemma 13.
The n-nested equivalences form an infinite family where $(\sim_{(n+1)S}) \subseteq (\sim_{nS})$. Also all these equivalences are distinct, that is $(\sim_{(n+1)S}) \neq (\sim_{nS})$.

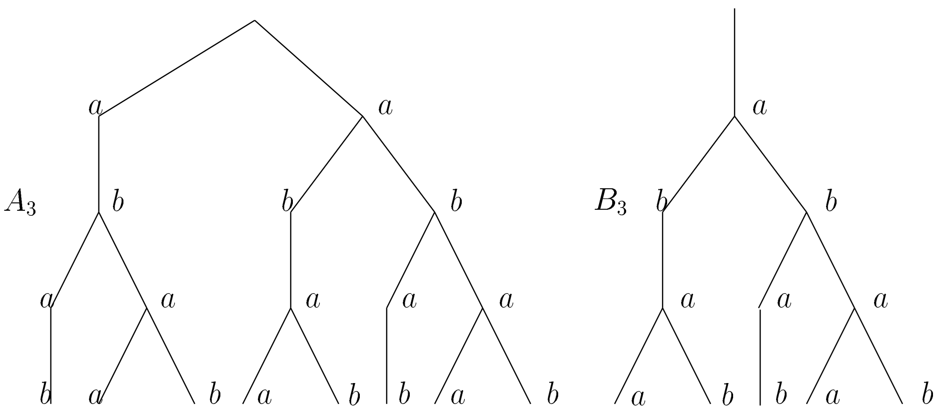
Proof
We know immediately that $\mathsf{f}_{(n+1)S} \longrightarrow \mathsf{f}_{nS}$, because of proposition 9. Therefore

$$P \sim_{(n+1)S} Q \Leftrightarrow P \models \mathsf{f}_{(n+1)S}(Q) \Rightarrow P \models \mathsf{f}_{nS}(Q) \Leftrightarrow P \sim_{nS} Q$$

To prove that the equivalences are distinct it is enough to construct a series of pairs of process terms A_n, B_n where $A_n \sim_{nS} B_n$, and $A_n \not\sim_{(n+1)S} B_n$. Let d_n be action name a when n is odd and b when n is even. Define

- $A_1 = a \cdot b \cdot 0 + a \cdot (b \cdot 0 + a \cdot 0)$, $B_1 = a \cdot (b \cdot 0 + a \cdot 0)$
- $A_{n+1} = d_{n+1} \cdot B_n + d_{n+1} \cdot A_n$
- $B_{n+1} = d_{n+1} \cdot A_n$.

Notice that the definition only requires two action names to construct A_n and B_n. If depicted as graphs, both A_n and B_n are binary trees. Here for example is A_3 and B_3.



The proof goes by induction on n. In the base case of $n = 2$ the result holds by explicit calculation.

Suppose then that $A_n \models \mathsf{f}_{nS} B_n$, and $A_n \not\models \mathsf{f}_{(n+1)S} B_n$. Let $d = d_{n+1}$, so $A_{n+1} = d \cdot B_n + d \cdot A_n$, $B_{n+1} = d \cdot A_n$.

By expansion

$$\mathsf{f}_{n+1} B_{n+1} = \Box(d \wedge \mathsf{f}_{n+1} A_n) \wedge \mathsf{f}'_n B_{n+1}$$

It is straightforward to see that $A_{n+1} \models \mathsf{f}'_n B_{n+1}$. Therefore to prove that $A_{n+1} \models \mathsf{f}_{n+1} B_{n+1}$ it is sufficient to prove that $A_{n+1} \models \Box(d \wedge \mathsf{f}_{n+1} A_n)$. This is true iff $d \cdot B_n \models \Box(d \wedge \mathsf{f}_{n+1} A_n)$, which is true iff $B_n \models \mathsf{f}_{n+1} A_n$. On expansion $\mathsf{f}_{n+1} A_n$ is

$$\Box(d_n \wedge \mathsf{f}_{n+1} B_{n-1} \vee d_n \wedge \mathsf{f}_{n+1} A_{n-1}) \wedge \mathsf{f}'_n A_n$$

From the induction hypothesis and the symmetry of $\sim_{nS}$, $B_n \models \mathsf{f}'_n A_n$. Recall that $B_n = d_n \cdot A_{n-1}$, therefore $B_n \models \Box(d_n \wedge \mathsf{f}_{n+1} A_{n-1})$. Clearly this implies

$$B_n \models \Box(d_n \wedge \mathsf{f}_{n+1} B_{n-1} \vee d_n \wedge \mathsf{f}_{n+1} A_{n-1})$$

Thus $A_{n+1} \models \mathsf{f}_{n+1} B_{n+1}$.

Expanding $\mathsf{f}'_{n+1} B_{n+1}$ gives

$$\mathsf{f}'_{n+1} B_{n+1} = \Diamond(d \wedge \mathsf{f}_{n+1} A_n) \wedge \mathsf{f}_n B_{n+1}$$

It is straightforward to prove $A_{n+1} \models \mathsf{f}_n B_{n+1}$, given that $A_n \models \mathsf{f}_{nS} B_n$ and therefore $B_n \models \mathsf{f}_{nS} A_n$. Since $A_{n+1} = d \cdot B_n + d \cdot A_n$, it is also straightforward to see that $A_{n+1} \models \Diamond(d \wedge \mathsf{f}_{n+1} A_n)$. Hence $A_{n+1} \models \mathsf{f}'_{n+1} B_{n+1}$. This completes the proof that $A_{n+1} \models \mathsf{f}_{(n+1)S} B_{n+1}$.

Suppose for a contradiction that $A_{n+1} \models \mathsf{f}_{(n+2)S} B_{n+1}$. By the expansion of f_{n+2} this implies that $A_{n+1} \models \Box(d \wedge \mathsf{f}_{n+2} A_n)$. Therefore, since $A_{n+1} = d \cdot B_n + d \cdot A_n$, $B_n \models \mathsf{f}_{n+2} A_n$. By proposition 9, f_{n+2} characterises forward f_{n+1}, and backward f'_{n+1} simulations, so that $B_n \models \mathsf{f}_{n+1} A_n \wedge \mathsf{f}'_{n+1} A_n$. This, by definition, implies $B_n \models \mathsf{f}_{(n+1)S} A_n$, which is a contradiction. This completes the proof of the lemma. □

The nS family is interesting because it converges to $\sim$.

Proposition 14.
Let $\sim_\sigma = \bigcap_{n \geq 1} (\sim_{nS})$. That is $P \sim_\sigma Q$ iff for every $n \geq 1$, $P \sim_{nS} Q$. The relation $\sim_\sigma$ is strong bisimulation equivalence $\sim$.

Proof
Define a relation $\sim_n$ recursively.

- $P \sim_{n+1} Q$ iff $\forall\ P \xrightarrow{a} P',\ \exists\ Q \xrightarrow{a} Q' : P' \sim_n Q'$, and $\forall\ Q \xrightarrow{a} Q',\ \exists\ P \xrightarrow{a} P' : Q' \sim_n P'$
- $P \sim_0 Q$ for any P and Q.

It is sufficient to prove that $P \sim_{nS} Q \Rightarrow P \sim_n Q$. This is proved by induction on n. The base case of $n = 1$ is easy.

Suppose that $P \sim_{nS} Q \Rightarrow P \sim_n Q$, and consider $P \sim_{(n+1)S} Q$. We know from proposition 9 that $\sim_{(n+1)S}$ is a forward and backward $\sim_{nS}$ simulation equivalence. Therefore if $P \sim_{(n+1)S} Q$ then

$$\forall\, P \xrightarrow{a} P',\ \exists\, Q \xrightarrow{a} Q' :\ P' \sim_{nS} Q'$$

By the induction hypothesis $P' \sim_{nS} Q' \Rightarrow P' \sim_n Q'$. Relation $\sim_{(n+1)S}$ is symmetric, from which it follows that

$$\forall\, Q \xrightarrow{a} Q',\ \exists\, P \xrightarrow{a} P' :\ Q' \sim_n P'$$

Hence $P \sim_{(n+1)S} Q \Rightarrow P \sim_{n+1} Q$. This completes the proof. $\Box$

The family nS are contrived solely for the purpose of constructing a filtration of $\sim$ into modal equivalences. The first in the family is S, simulation equivalence. This proposition proves that modal equivalences occur throughout the whole spectrum of simulation like equivalences, from simulation to bisimulation.

Next is defined a second infinite family, nRS. This is constructed by using the same idea which defined nS, but starting with ready simulation of section 3.2 instead of simulation. Define

$$e'(P) = \bigwedge\{\Diamond\, a \mid P \xrightarrow{a} P'\}$$
$$e(P) = \Box(\bigvee\{a \mid P \xrightarrow{a} P'\})$$

In section 3.2 these formulae are called g and h. We have renamed them here to avoid confusing them with the g and h used in the definition of the template in section 3.8. Recall that $P \models e'(Q)$ implies that $I(Q) \subseteq I(P)$, and $P \models e(Q)$ implies that $I(P) \subseteq I(Q)$.

Ready simulation has the characteristic formula $\mathsf{f}_{RS} = \mathsf{rf} \wedge \mathsf{rf}'$ where

- $\mathsf{rf}(P) = \Box(\bigvee\{a \wedge \mathsf{rf}(P') \mid P \overset{a}{\longmapsto} P'\}) \wedge e'(P)$
- $\mathsf{rf}(0) = \Box\, \mathsf{F}$
- $\mathsf{rf}'(P) = (\bigwedge\{\Diamond(a \wedge \mathsf{rf}'(P')) \mid P \overset{a}{\longmapsto} P'\}) \wedge e(P)$
- $\mathsf{rf}'(0) = \mathsf{T}$

Define $\mathsf{rf}_{(n+1)}$, and $\mathsf{rf}'_{(n+1)}$ by the abbreviated equations

- $\mathsf{rf}_{n+1}(P) = \Box(\bigvee\{a \wedge \mathsf{rf}_{(n+1)}(P') \mid P \overset{a}{\longmapsto} P'\}) \wedge e'(P) \wedge \mathsf{rf}'_n(P)$
- $\mathsf{rf}_{n+1}(0) = \Box\, \mathsf{F}$
- $\mathsf{rf}'_{n+1}(P) = \bigwedge\{\Diamond(a \wedge \mathsf{rf}'_{n+1}(P')) \mid P \overset{a}{\longmapsto} P'\} \wedge e(P) \wedge \mathsf{rf}_n(P)$

- $\mathsf{rf}'_{n+1}(0) = \mathsf{T}$

Define $\mathsf{rf}_1 = \mathsf{rf}$, and $\mathsf{rf}'_1 = \mathsf{rf}'$. Define $\mathsf{f}_{nRS} = \mathsf{rf}_n \wedge \mathsf{rf}'_n$. Notice that $\mathsf{f}_{1RS} = \mathsf{f}_{RS}$.

Proposition 15.
For each n, $(n+1)S \subsetneqq nRS \subsetneqq nS$.

Proof
In [6] it is proved that $2S \subsetneqq RS$. It is also easy to see that $\mathsf{f}_{2S} \rightarrow \mathsf{f}_{RS}$. A straightforward induction on n, proves that $\mathsf{f}_{(n+1)S} \rightarrow \mathsf{f}_{nRS}$. It is trivially the case that $\mathsf{f}_{nRS} \rightarrow \mathsf{f}_{nS}$.

Define a series of pairs C_n, D_n where $C_n \sim_{nRS} D_n$ and $C_n \not\sim_{(n+1)S} D_n$ with the recursive equations

- $C_1 = a \cdot b \cdot c \cdot 0 + a \cdot (b \cdot c \cdot 0 + c \cdot d \cdot 0)$, $D_1 = a \cdot (b \cdot c \cdot 0 + c \cdot d \cdot 0)$
- $C_{n+1} = d_{n+1} \cdot D_n + d_{n+1} \cdot C_n$
- $D_{n+1} = d_{n+1} \cdot C_n$.

Where $d_{2m} = b$, $d_{2m+1} = a$. Just as with the proof of lemma 13, one can prove that $C_n \models \mathsf{f}_{nRS} D_n$, and that $C_n \not\models \mathsf{f}_{(n+1)S} D_n$. Hence $(n+1)S \neq nRS$.

The series A_n, B_n defined in the proof of lemma 13 is enough to distinguish between nS and nRS. In an analogous fashion to the proof of lemma 13, one can prove that $A_n \models \mathsf{f}_{nS} B_n$ and $A_n \not\models \mathsf{f}_{nRS} B_n$. Thus $nS \neq nRS$. This completes the proof. $\square$

This proposition shows that there is an infinite family which fits in between the members of the nS family. Of course since $(n+1)S \subsetneqq nRS \subsetneqq nS$, it is immediate that nRS converges to $\sim$. This evidence suggests there are a lot of ways to divide up the simulation/bisimulation spectrum into distinct infinite families of modal equivalences.

The remainder of this section shows that $\approx$ and $=_\tau$ are also the limit of infinite families of modal equivalences. Define

- $w\mathsf{f}(P) = \Box(\bigvee\{\Box^*_a\, w\mathsf{f}(P') \mid P \overset{a}{\longmapsto} P'\})$, $w\mathsf{f}(0) = \Box_\tau \mathsf{F}$
- $w\mathsf{f}'(P) = (\bigwedge\{\Diamond\, \Diamond^*_a\, w\mathsf{f}_h(P') \mid P \overset{a}{\longmapsto} P'\})$, $w\mathsf{f}'(0) = \mathsf{T}$.
- $w\mathsf{f}_{n+1}(P) =$
  $\Box(\bigvee\{\Box^*_a\, w\mathsf{f}_{(n+1)}(P') \mid P \overset{a}{\longmapsto} P'\}) \wedge w\mathsf{f}'_n(P)$, $w\mathsf{f}_{n+1}(0) = \Box_\tau \mathsf{F}$
- $w\mathsf{f}'_{n+1}(P) =$
  $\bigwedge\{\Diamond\, \Diamond^*_a\, w\mathsf{f}'_{n+1}(P') \mid P \overset{a}{\longmapsto} P'\} \wedge w\mathsf{f}_n(P)$, $w\mathsf{f}'_{n+1}(0) = \mathsf{T}$

Define $w\mathsf{f}_1 = w\mathsf{f}$, $w\mathsf{f}'_1 = w\mathsf{f}'$, and $w\mathsf{f}_{nS} = w\mathsf{f}_n \wedge w\mathsf{f}'_n$. This defines an infinite family of modal equivalences $\approx_{nWS}$ where $P \approx_{nWS} Q$ iff $P \models w\mathsf{f}_{nS}(Q)$. Each nWS is a weak forward $(n-1)WS$ simulation equivalence and weak backward $(n-1)WS$ simulation equivalence, also nWS is defined in accordance with the weak template of the last section. The results about nS have their counterparts for nWS, and the proofs for nWS are analogous to the proofs for nS. The important result about the nWS family is that it converges to weak bisimulation.

Proposition 16.
Let $(\approx_\sigma) = \bigcap_{n\geq 1}(\approx_{nWS})$. That is $P \approx_\sigma Q$ iff for every $n \geq 1$, $P \approx_{nWS} Q$. The relation $\approx_\sigma$ is weak bisimulation equivalence $\approx$.

Finally in this section we define a family which converges to congruence. Given the family nWS, and the fact that f_τ is defined directly from $\mathsf{f}_\approx$, it is not surprising to find that there is an infinite family converging to $=_\tau$ defined directly out of nWS. Define f_τ^n by the abbreviated equations:

- $$\begin{array}{rcl} \mathsf{f}_\tau^n(P) & = & \Box(\bigvee\{a \wedge w\mathsf{f}_{nS}(P') \mid P \stackrel{a}{\longmapsto} P'\}) \\ & & \wedge \\ & & \bigwedge\{\Diamond(a \wedge w\mathsf{f}_{nS}(P')) \mid P \stackrel{a}{\longmapsto} P'\} \end{array}$$
- $\mathsf{f}_\tau^n(0) = \Box\,\mathsf{F}$.

It is easy to see that $\lim \mathsf{f}_\tau^n = \mathsf{f}_\tau$, since $\lim w\mathsf{f}_{nS} = \mathsf{f}_\approx$. The following proposition rephrases this observation more formally. Let $P =_\tau^n Q$ be defined by $P \models \mathsf{f}_\tau^n(Q)$.

Proposition 17.
Let $(=_\sigma) = \bigcap_{n \geq 1}(=_\tau^n)$. That is $P =_\sigma Q$ iff for every $n \geq 1$, $P =_\tau^n Q$. The relation $=_\sigma$ is congruence $=_\tau$.

5 Trace Like Equivalences

So far the paper has been solely concerned with equivalences which lie between simulation and bisimulation. This section shows constructively that some well known trace like equivalences are logic equivalences.

Definition 18.
An observational equivalence relation $\sim_\sigma$ over processes is a logic equivalence if for each P there is a formula $\mathsf{f}_\sigma^P \in \mu\mathbb{K}$ such that

$$P \sim_\sigma Q \text{ iff } \models \mathsf{f}_\sigma^P \leftrightarrow \mathsf{f}_\sigma^Q$$

The introduction proved that any observation equivalence is a logic equivalence by applying the axiom of choice. This section demonstrates that a formula for f_σ^P can be effectively constructed for trace, complete trace, failure trace, ready trace and possible future equivalence.

5.1 Trace and Complete Trace Equivalence

Take the sequence $\eta = a_1 a_2 \cdots$ where the a_i are action names. Write $P \models \eta$ iff there is a sequence of transitions

$$P \xrightarrow{a_1} P_1 \xrightarrow{a_2} P_2 \xrightarrow{a_2} \cdots$$

Write $P \sim_{tr} Q$ iff for every sequence η, $P \models \eta \Leftrightarrow Q \models \eta$ This relation is as old as time itself. It can be found most easily in [14].

A process P is deterministic if $P \xrightarrow{a} P'$ and $P \xrightarrow{a} P''$ implies that $P' \sim P''$. In chapter 10 of [16] Milner points out that if two processes P, Q are deterministic, then $P \sim_{tr} Q$ iff $P \sim Q$. It is equally true that for any equivalence σ between trace and bisimulation, that $P \sim_{tr} Q$ iff $P \sim_\sigma Q$ iff $P \sim Q$.

From P one may consruct a deterministic process P^* which is trace equivalent to P. Recall that we regard an image finite process P as equal to the edge labeled graph which depicts it. Suppose there are a pair of nodes v_1, v_2 in P such that for some $u \in N(P)$ and $a \in A$,

$u \xrightarrow{a} v_1 \in P$ and $u \xrightarrow{a} v_2 \in P$. Glue together v_1 and v_2. That is, first remove v_2. Then for any edge $w \xrightarrow{b} v_2 \in P$, add an edge $w \xrightarrow{b} v_1$. For any edge $v_2 \xrightarrow{b} w \in P$, add an edge $v_1 \xrightarrow{b} w$. When this procedure is applied exhaustively to P the resultant graph depicts a deterministic process P^* which is trace equivalent to P.

Suppose that $\sim_\sigma$ is any modal equivalence. It is therefore between trace and bisimulation. Consider $\mathsf{f}_\sigma(P^*)$. Because P^* is deterministic the whole spectrum of observational equivalences has collapsed, so that $P^* =_{tr} Q^*$ iff $P^* \sim_\sigma Q^*$. Hence we may define $\mathsf{f}^P_{tr} = \mathsf{f}_\sigma(P^*)$.

The formula f_S is the smallest possible characteristic formula for a modal simulation equivalence. It makes sense therefore to define $\mathsf{f}^P_{tr} = \mathsf{f}_S(P^*)$.

Complete Trace

Redefine $P \models \eta$ for the case of finite η. If $\eta = a_1 a_2 \cdots a_n$, then $P \models \eta$ iff there is a sequence of transitions

$$P \xrightarrow{a_1} P_1 \xrightarrow{a_2} P_2 \xrightarrow{a_2} \cdots \xrightarrow{a_n} 0$$

The definition of $P \models \eta$ for the case of infinite η is unchanged. Define $P \sim_{Ctr} Q$ if for every trace (finite or infinite) η, $P \models \eta \Leftrightarrow Q \models \eta$.

In this case, as one would expect, one can define $\mathsf{f}^P_{Ctr} = \mathsf{f}_{CS}(P^*)$. See section 3.3 for the definition of f_{CS}.

5.2 Failure Trace Equivalence

Take the sequence $\eta = a_1 X_1 a_2 X_2 \cdots$ where the a_i are action names and the X_i are finite sets of action names. Write $P \models \eta$ iff there is a sequence of transformations

$$P \xrightarrow{a_1} P_1 \xrightarrow{a_2} P_2 \xrightarrow{a_2} \cdots$$

and for each i, and each $x \in X_{i+1}$ there is no transformation of the form $P_i \xrightarrow{x} P'$ (regard P as P_0). Write $P =_{Ftr} Q$ iff for every sequence η, $P \models \eta \Leftrightarrow Q \models \eta$ This relation also goes by the name of refusal semantics [21]. It is an obvious derivative of failure semantics [3].

This is a different equivalence to trace, for example

$$a \cdot (b + c) \neq_{Ftr} a \cdot b + a \cdot c$$

whereas $a \cdot (b + c) =_{tr} a \cdot b + a \cdot c$.

For a process P and node $u \in N(P)$ let $V(P)$ be the set of action names which appear in P. Let $X(P) = V(P) - I(P)$, where $I(P)$ is defined in section 3.2. Create a graph P^* from P as follows. Suppose there is a pair of nodes v_1, v_2 in P such that for some $u \in N(P)$ and $a \in A$, $u \xrightarrow{a} v_1 \in P$, $u \xrightarrow{a} v_2 \in P$ and $X(v_1) \subseteq X(v_2)$. Add an edge $v_1 \xrightarrow{b} w$ for each edge $v_2 \xrightarrow{b} w \in P$. Process P^* is constructed by exhaustively applying this procedure to P.

The process P^* is failure trace equivalent to P. This is true because the above procedure preserves failure trace equivalence. Process P^* is unique up to strong bisimulation. Thus for any equivalence σ between failure trace and strong bisimulation $P^* \sim_\sigma Q^*$ iff $P^* \sim_{Ftr} Q^*$. Define $\mathsf{f}^P_{Ftr} = \mathsf{f}_S(P^*)$, one can prove $P \sim_{Ftr} Q$ iff $\models \mathsf{f}_S(P^*) \leftrightarrow \mathsf{f}_S(Q^*)$.

5.3 Ready Trace Equivalence

This equivalence has occurred in the literature under various guises. See [22], [1] or [23] for example.

Take $\eta = a_1 X_1 a_2 X_2 \cdots$ where the a_i are action names and the X_i are finite sets of action names. Write $P \models \eta$ iff there is a sequence of transformations

$$P = P_0 \xrightarrow{a_1} P_1 \xrightarrow{a_2} P_2 \xrightarrow{a_2} \cdots$$

and for each i, $X_{i+1} = I(P_i)$. Write $P =_{Rtr} Q$ iff for every sequence η, $P \models \eta \Leftrightarrow Q \models \eta$

This is a different equivalence to trace, and failure trace for example

$$a \cdot (b + c) + a \cdot b + a \cdot c \neq_{Rtr} a \cdot b + a \cdot c$$

whereas the two terms are failure trace equivalent.

Create a graph P^* from P as follows. Suppose there is a pair of nodes v_1, v_2 in P such that for some $u \in N(P)$ and $a \in A$, $u \xrightarrow{a} v_1 \in P$, $u \xrightarrow{a} v_2 \in P$ and $I(v_1) = I(v_2)$. Glue together v_1 and v_2 (just as in the definition of P^* for trace). Exhaustively applying this procedure constructs P^*. Define $\mathsf{f}^P_{Rtr} = \mathsf{f}_S(P^*)$. One can show that $P \sim_{Rtr} Q$ iff $\models \mathsf{f}_S(P^*) \leftrightarrow \mathsf{f}_S(Q^*)$.

5.4 Possible Future

Possible future equivalence [24] could accurately be described as 2-nested trace equivalence. Let $T(P) = \{\eta \mid P \models \eta\}$, where $P \models \eta$ is given the same definition as for trace equivalence.

Let η be a sequence $a_1 a_2 \cdots a_n$, and let X be a set of traces. Define $P \models_* \eta X$ if there is a sequence of transitions

$$P \xrightarrow{a_1} P_1 \xrightarrow{a_2} P_2 \xrightarrow{a_2} \cdots \xrightarrow{a_n} P_{n+1}$$

where $T(P_{n+1}) = X$. In the case of infinite η, $P \models_* \eta X$ just when $P \models \eta$. Let $Pf(P)$ be the set of ηX such that $P \models_* \eta X$. Define $P \sim_{Pf} Q$ iff $Pf(P) = Pf(Q)$.

Create a graph P^* from P as follows. Suppose there is a pair of nodes v_1, v_2 in P such that for some $u \in N(P)$ and $a \in A$, $u \xrightarrow{a} v_1 \in P$, $u \xrightarrow{a} v_2 \in P$ and $T(v_1) \subseteq T(v_2)$. Then glue together v_1 and v_2 (just as in the definition of P^* for trace). Exhaustively applying this procedure constructs P^*. Define $\mathsf{f}^P_{Pf} = \mathsf{f}_S(P^*)$. One can show that $P \sim_{Pf} Q$ iff $\models \mathsf{f}_S(P^*) \leftrightarrow \mathsf{f}_S(Q^*)$.

Notice that $T(v_1) \subseteq T(v_2)$ iff $\mathsf{f}^{v_1}_{tr} \longrightarrow \mathsf{f}^{v_2}_{tr}$. Thus the construction P^* can be described in terms of the logic $\mu\mathbb{K}$. Thinking of pure future as 2-nested trace equivalence leads to the generalised concept of n-nested trace equivalence.

Suppose that n-nested trace equivalence is defined and is a logic equivalence with formulae f^P_{ntr}. Consider a process P^*_{n+1} defined recursively as follows. Start with P, Suppose there is a pair of nodes v_1, v_2 in P such that for some $u \in N(P)$ and $a \in A$, $u \xrightarrow{a} v_1 \in P$, $u \xrightarrow{a} v_2 \in P$ and $\mathsf{f}^{v_1}_{ntr} \longrightarrow \mathsf{f}^{v_2}_{ntr}$. Then glue together v_1 and v_2. Exhaustively applying this procedure constructs P^*_{n+1}. Define $\mathsf{f}^P_{(n+1)tr} = \mathsf{f}_S(P^*_n)$. Define $P \sim_{(n+1)tr} Q$ iff $\models \mathsf{f}^P_{(n+1)tr} \leftrightarrow \mathsf{f}^Q_{(n+1)tr}$. This generates an infinite family of trace like equivalences which are logic equivalences and whose limit is the simulation equivalence $\sim_S$.

6 Conclusion

The operational definition of an observational equivalence does not usually give any great insight into the meaning encoded by the equivalence. The trend over the last ten years has been to describe the denotation of an equivalence with some kind of none standard logic, such as the Hennessey Milner logic. Each equivalence σ is described by its own specific logic L_σ. For a process P its σ equivalence class is represented by the set of logic formulae $L_\sigma(P)$ which P satisfies when regarded as a model in the logic L_σ. Then $P \sim_\sigma Q$ iff $L_\sigma(P) = L_\sigma(Q)$.

In this paper the philosophy is to use a single logic, $\mu\mathbb{K}$, on which to base the semantics of every observational equivalence. The paper proves that every observational equivalence is a logic equivalence. That is, for each σ and each P there is a single formula $\mathsf{f}_P^\sigma \in \mu\mathbb{K}$ which is the characteristic formula for P with respect to σ.

The paper also proves many equivalences σ are modal equivalences and therefore can be given a characteristic formula $\mathsf{f}_\sigma \in \mathbb{K}^{Par}$. For any particular process P, $\mathsf{f}_\sigma(P) \in \mu\mathbb{K}$. For a general logic equivalence σ, the denotation of σ is effectively the set of formulae f_P^σ. For a modal equivalence σ the denotation is a single formula f_σ.

The paper proves there are infinitely many modal equivalences which are 'sensible' simulation equivalences. (A 'sensible' equivalence is one which is a forward g simulation equivalence, and a backward h simulation equivalence for suitable g and h). Moreover there are infinite families of modal equivalences which converge to $\sim$, $\approx$ and $=_\tau$, each of which is also a modal equivalence. This shows there is a large body of modal equivalences which contain many significant examples.

References

[1] J. C. M. Baeten, J. A. Berstra, J. W. Klop, 'Ready Trace semantics for concrete algebra with the priority operator', the Computer Journal, 1987, 30, vol 6, pp498-506.

[2] B. Bloom, S. Istrail, A. R. Meyer, 'Bisimulation can't be traced: preliminary report', POPL'15, 15th ACM Symposium on Principles of Progamming Languages 1988, pp 229-239.

[3] S. D. Brookes, C. A. R. Hoare, A. W. Roscoe, 'A theory of communicating sequential processes', JACM 31, vol 3, pp. 560-599.

[4] M. Dam, 'Model Checking Mobile Processes', LNCS 715, 1993, pp. 22-36.

[5] S. Graf, J.Sifakis, 'A modal characterisation of observational equivalence of finite terms of CCS', Information and Control 68, 125-145, 1986.

[6] R. J. van Glabbeek, 'The Linear Time - Branching Time Spectrum', Concur'90, LNCS 458, pp. 278-297.

[7] R. J. van Glabbeek, 'The Linear Time - Branching Time Spectrum II', Concur'93, LNCS 715.

[8] U. Goltz, Ruurd Kuiper, W. Penczek, 'Propositional Temporal Logic and Equivalences', Concur'92, LNCS 630.

[9] J. F. Groote, F. W. Vaandrager, 'Structured operational semantics and bisimulation as a congruence', Report CS-R8845, CWI, Amsterdam, also an extended abstract in ICALP 89, LNCS 372, pp423-438.

[10] M. Hennessey 'A Model for the π-Calculus, Technical Report, Computer Science Dept, Sussex University 1992.

[11] M. Hennessey, R. Milner 'On observing nondeterminism and concurrency', Springer-Verlag LNCS 85, 295-309, 1980.

[12] M. Hennessey, R. Milner 'Algebraic Laws for Nondeterminism and Concurrency', JACM 32(1), pp. 137-161.

[13] M. Hennessey, X. Liu 'A Modal Logic for Message Passing Processes', Technical Report, Computer Science Dept, Sussex University January 1993.

[14] C. A. R. Hoare 'Communicating Sequential Processes' Prentice Hall, 1985.

[15] G. E. Hughes, M.J.Cresswell, 'An Introduction to Modal Logic', Methuen(UP) 1969.

[16] R. Milner 'Communication and Concurrency', Prentice Hall, 1989.

[17] R. Milner J. Parrow, D. Walker 'A Calculus of Mobile Processes' I and II, Information and Computation, V100, pp 1-78, 1992.

[18] R. Milner 'Action Structures', Technical Report, Computer Science Dept, Edinburgh University, December 1992.

[19] W. P. R. Mitchell 'Modal Logic and Equality for Process Algebra', to appear in J. Logic and Computation, Vol 5 1995.

[20] D. M. R. Park, 'Concurrency and automata on infinite sequences', Proceedings of 5th GI Conference, LNCS 104, pp 167-183.

[21] I. C. C. Phillips 'Refusal Testing', journal of Theoretical Computer Science, 50, pp. 241-284.

[22] A. Pnueli, 'Linear and branching structures in the semantics and logics of reactive systems', ICALP'85, LNCS 194, pp. 15-32.

[23] L. Pomello, 'Some equivalence notions for concurrent systems. An overview', Advances in Petri Nets 1985, LNCS 222, pp 381-400.

[24] W. C. Rounds, S. D. Brookes, 'Possible Futures, acceptances, refusals, and communicating processes', Proceedings 22nd Annual Symposium on Foundations of Computer Science, 1982 IEEE, pp. 140-149.

[25] D. Walker 'Logics for Processes', Proceedings of the VI Banff Workshop: Process Algebras, Banff, Canada, August 1992.