Operational semantics, bisimulations and logical complexity

R.T.P. Fernando

Computer Science/Department of Software Technology

Report CS-R9355 August 1993

CWI is the National Research Institute for Mathematics and Computer Science. CWI is part of the Stichting Mathematisch Centrum (SMC), the Dutch foundation for promotion of mathematics and computer science and their applications.
SMC is sponsored by the Netherlands Organization for Scientific Research (NWO). CWI is a member of ERCIM, the European Research Consortium for Informatics and Mathematics.

P.O. Box 94079, 1090 GB Amsterdam (NL)
Kruislaan 413, 1098 SJ Amsterdam (NL)
Telephone +31 20 592 9333
Telefax +31 20 592 4199

Operational Semantics, Bisimulations and Logical Complexity

Tim Fernando

CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands
fernando@cwi.nl

Abstract

A logical analysis of operational semantics based on the notion of a bisimulation is considered, and the explosion in logical complexity triggered by bisimulations on computable transition predicates is investigated in relation to certain algebraic styles of reasoning. A 'universal mechanical' transition predicate is isolated, accomodating transition predicates departing in essential ways from those underlying various models from denotational semantics. Objections commonly raised against bisimulations (e.g., that bisimilarity is too fine, or that it may fail to be a congruence, or that it is based on transition graphs reducing parallelism to interleaving) are disputed, resting (as these do) on unfortunate choices of transition predicates. For while a bisimulation presupposes a fixed transition predicate, there are a multitude of such predicates, representing (for instance) different levels of abstraction, including input/output equivalence, which, in the context of partial equivalence relations, is presented (below) as a bisimulation.

AMS Subject Classification (1991): 03B70, 03D45, 68Q55
CR Subject Classification (1991): D.3.1, F.3.2, F.4.1
Keywords and Phrases: operational semantics, labelled transition systems, bisimulations, logical complexity, partial equivalence relations
Note: This work was funded by the Netherlands Organization for Scientific Research (NWO Project NF 102/62-356, 'Structural and Semantic Parallels in Natural Languages and Programming Languages').

As popular as the notion of a bisimulation (Park [20]) has become, various points about it have troubled researchers working on operational semantics (and/or process theory). Two important problems facing an algebraic approach are

(A) the question of whether or not bisimulation equivalence — bisimilarity, for short — is a congruence relative to some set of functions on states,

and

(B) the relationship between bisimilarity and and its finitary co-inductive approximation (called observational equivalence in Hennessy and Milner [15]).

A problem related to (B) is the mechanical uncomputability of bisimilarity over certain transition predicates, the typical response to which has been to focus attention on isolating special predicates over which bisimilarity is computationally tractable. Consequently, the study of bisimulations over arbitrary mechanically generated transitions has largely been neglected. The present work is an attempt to correct this oversight, the basic thesis being that the interest and scope of bisimulations rest in no small measure on their astronomical complexity (arising from mechanical computation). With regard to (B), we show how model constructions based on compactness (e.g., Stone duality) introduce 'ideal' elements that reconcile bisimilarity with its finitary approximation. (Unfortunately, although such constructions can be arranged to respect first-order logic, bisimilarity is not a first-order notion.) And as for (A), we describe how to *turn* bisimilarity into a congruence (if it is not already one).

In addition, the following technical results exploring the logical complexity of bisimulations are presented, building on Fernando [9], where bisimilarity is proved to fall outside Π^1_1 relative to a certain transition predicate $\succ$ (with one label) computable in logspace.

Report CS-R9355
ISSN 0169-118X
CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

(i) Let $\rightarrow$ be bisimilar to $\succ$. If bisimilarity relative to $\rightarrow$ is Π^1_1, then $\rightarrow$ is *not* Σ^1_1.

(ii) There is a 'universal' mechanical transition predicate $\rightsquigarrow$. The result $\rightsquigarrow/\underline{\leftrightarrow}$ of dividing $\rightsquigarrow$ by bisimilarity $\underline{\leftrightarrow}$, defined by

$$s \overset{l}{\rightsquigarrow}/\underline{\leftrightarrow}\, t \quad \text{iff} \quad (\exists s', t')\ s \underline{\leftrightarrow} s' \ \&\ t \underline{\leftrightarrow} t' \ \&\ s' \overset{l}{\rightsquigarrow} t' \ ,$$

is Σ^1_1-complete under many-one reductions.

Finally, contrary to the claim that bisimilarity is too fine as an equivalence, we exhibit how bisimulations generalize input/output equivalence over hereditarily effective operations (Kreisel [16]) by forming transition predicates relative to which the equalities become bisimulations. The aim is to provide a smooth generalization of mechanical input/output statements to transitions, respecting cartesian types (i.e., a fixed notion of application).

1 Background definitions and facts

The present section reviews some well-known material, found, for example, in Milner [17]. A *(labelled) transition system* is a triple $(L, S, \rightarrow)$ where $\rightarrow \subseteq S \times L \times S$. L is said to be the set of *labels*, S the set of *states*, $\rightarrow$ the *transition predicate*, and a *transition* $(s, l, s') \in \rightarrow$ is written $s \overset{l}{\rightarrow} s'$. Given transition systems $(L, S, \rightarrow)$ and $(L, S', \rightarrow')$ over the same label set L, a relation $R \subseteq S \times S'$ is a *bisimulation* if whenever sRs' then for all $l \in L$,

$$(\forall t \overset{l}{\leftarrow} s)(\exists t' \overset{l}{\leftarrow'} s')\ tRt' \quad \text{and} \quad (\forall t' \overset{l}{\leftarrow'} s')(\exists t \overset{l}{\leftarrow} s)\ tRt' \ .$$

States s and s' are said to be *bisimilar*, which we write as $s \underline{\leftrightarrow} s'$, if there is a bisimulation relating s to s'. Note that the relation $\underline{\leftrightarrow}$ of *bisimilarity* is a bisimulation iff there is a largest bisimulation (in the sense of $\subseteq$). It turns out that there is a largest bisimulation. This can be seen by rephrasing the definition of a bisimulation as follows: $R \subseteq S \times S'$ is a *bisimulation* if $R \subseteq R^{bf}$, where $\cdot^{bf}$ is the "back-and-forth" operator on binary relations defined by

$$R^{bf} \ = \ \{(s, s') \in S \times S' \ \mid \ (\forall l \in L) \ \ (\forall t \overset{l}{\leftarrow} s)(\exists t' \overset{l}{\leftarrow'} s')\ tRt' \ \text{ and } \ (\forall t' \overset{l}{\leftarrow'} s')(\exists t \overset{l}{\leftarrow} s)\ tRt'\} \ .$$

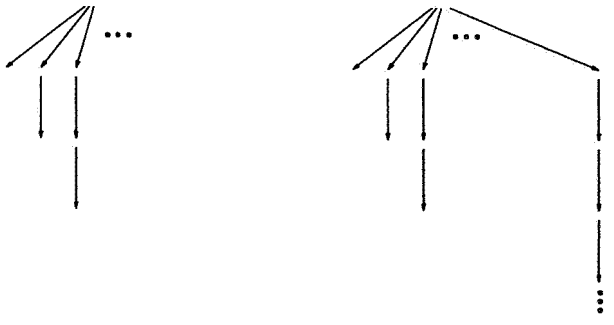
Now, because R occurs only positively in R^{bf}, the map $R \mapsto R^{bf}$ is ($\subseteq$-)monotone and the largest bisimulation $\underline{\leftrightarrow}$ can be calculated co-inductively as $\bigcap \{\underline{\leftrightarrow}_\alpha \mid \alpha < \mathrm{card}(S \times S')^+\}$, where

$$\underline{\leftrightarrow}_\alpha \ = \ \bigcap_{\beta < \alpha} (\underline{\leftrightarrow}_\beta)^{bf} \ .$$

Observe that $S \times S' = \underline{\leftrightarrow}_0 \supseteq \underline{\leftrightarrow}_1 \supseteq \cdots \supseteq \underline{\leftrightarrow}_\omega \supseteq \underline{\leftrightarrow}_{\omega+1} \supseteq \cdots \supseteq \underline{\leftrightarrow}$.

1.1 A notorious counter-example

The relation $\underline{\leftrightarrow}_\omega = \bigcap_{n<\omega} \underline{\leftrightarrow}_n$ is a conjunction of "finitary" predicates $\underline{\leftrightarrow}_n$, which is to say, $\underline{\leftrightarrow}_\omega$ can be captured by a finitary formal language, as spelled out in Hennessy and Milner [15]. The equivalence $\underline{\leftrightarrow}_\omega$ need not, however, coincide with $\underline{\leftrightarrow}$, as demonstrated by



where $|L| = 1$. Confusing these two transition systems is problematic inasmuch as the infinite branch can be interpreted as a failure of termination, or say, an unfair merging of an infinite stream 1^∞ of 1's with 0. It is easy enough to repair the Hennessy-Milner characterization [15] (so as to apply to $\underline{\leftrightarrow}$ in general) by allowing infinitary disjunctions (and conjunctions), but then the problem arises as to what is taken for granted by building these infinitary constructs into the logic. (That is, a free-wheeling introduction of infinitary notions begs the question of effectiveness. Note also that such constructs spoil compactness, which is crucial to Abramsky [1].)

1.2 Rooted transition systems

Transition systems typically have obvious "roots" (or "initial" states) $s_0 \in S$ and $s'_0 \in S'$, in which case we express the assertion $s_0 \underline{\leftrightarrow} s'_0$ by saying that the *rooted* (or *pointed*) transition systems $(L, S, \rightarrow, s_0)$ and $(L, S', \rightarrow', s'_0)$ are *bisimilar*, again writing $(L, S, \rightarrow, s_0) \underline{\leftrightarrow} (L, S', \rightarrow', s'_0)$.

2 Two basic results on logical complexity

We present model-theoretic and recursion-theoretic (or, as formulated in Girard [11], proof-theoretic) arguments demonstrating the inadequacy of a certain algebraic style of reasoning advocated, for example, in Baeten and Weijland [3]. That style of reasoning consists of an equational system and an infinitary induction principle. The former obviously forms a modest fragment of first-order logic, while the latter amounts similarly to a minute bite out of what is called Π^1_1-logic in Girard [11]. We begin by demonstrating how the infinitary induction principle is validated by first-order tricks, but then establish the inadequacy of Π^1_1-logic, let alone first-order logic.

2.1 Observational equivalence and 'ideal' elements

The *Approximation Induction Principle* (AIP) of Baeten and Weijland [3] is the infinitary rule

$$\frac{(p)_0 = (q)_0 \qquad (p)_1 = (q)_1 \qquad \cdots}{p = q} \,,$$

where the nth *projection* $(p)_n$ of p is the approximation of p up to n transitions, given by

$$\frac{p \xrightarrow{l} q}{(p)_{n+1} \xrightarrow{l} (q)_n} \,.$$

Interpreting $=$ as $\underline{\leftrightarrow}$, AIP says the coinductive approximation of $\underline{\leftrightarrow}$ converges at ω, since

$$(p)_n \underline{\leftrightarrow} (q)_n \quad \text{iff} \quad p \underline{\leftrightarrow}_n q \,.$$

Just how badly this rule fails is a measure of the complexity of $\underline{\leftrightarrow}$ (e.g., Moschovakis [19]). This point underlies the arguments in the next subsection, but for now, let us observe how this rule is validated algebraically.

It is clear enough (from, for example, Hennessy and Milner [15]) that AIP is sound for finitely branching (or even so-called image finite) transition systems. What we show now is why it tends to hold in "large" models constructed algebraically. Towards this end, let us regard a transition system $(L, S, \rightarrow)$ as a first-order model $(L \cup S, L, \rightarrow)$ in the language with a unary relation symbol $\dot{L}$ for labels, and a 3-ary relation symbol T for transitions. Note that $\underline{\leftrightarrow}_\omega$ on $(L, S, \rightarrow)$ can be defined by the conjunction of an infinite set of first-order $\{\dot{L}, T\}$-definable predicates $\sim_n$ (for every $n < \omega$) expressing $\underline{\leftrightarrow}_n$. Recall (or else consult, for example, Sacks [22]) that a set $\Phi(x)$ of first-order formulas $\varphi(x)$ with at most one free variable x is *satisfiable* (or *realized*) *in* M if there is an element m of M such that for every $\varphi(x)$ in $\Phi(x)$, $M \models \varphi[m]$. $\Phi(x)$ is *finitely satisfiable in* M if every finite subset of $\Phi(x)$ is satisfiable in M. Given a model M (often fixed and suppressed in the background), a *type* is a finitely satisfiable set of formulas.

Proposition 1. *AIP (i.e., $\underline{\leftrightarrow} = \underline{\leftrightarrow}_\omega$) is a statement about certain types being realized. More precisely, $\underline{\leftrightarrow} = \underline{\leftrightarrow}_\omega$ in $(L \cup S, L, \rightarrow)$ iff for all s, s', t and l such that $s' \xrightarrow{l} t$, every set*

$$\Phi_{s,t,l}(x) \quad = \quad \{T(\dot{s}, \dot{l}, x)\} \ \cup \ \{\dot{t} \sim_n x \mid n < \omega\}$$

of $\{\dot{L}, T, \dot{s}, \dot{t}, \dot{l}\}$-formulas that is finitely satisfiable in the $\{\dot{L}, T, \dot{s}, \dot{t}, \dot{l}\}$-model $(L \cup S, L, \rightarrow, s, t, l)$ is satisfiable there.

Proof. As $\underline{\leftrightarrow} = \underline{\leftrightarrow}_\omega$ is equivalent to $\underline{\leftrightarrow}_\omega \subseteq \underline{\leftrightarrow}_{\omega+1}$, it suffices to observe that for all $s, s' \in S$,

(i) $s \underline{\leftrightarrow}_\omega s'$ iff whenever $s' \xrightarrow{l} t$, $\Phi_{s,t,l}(x)$ is finitely satisfiable in $(L \cup S, L, \rightarrow, s, t, l)$, and

(ii) $s \underline{\leftrightarrow}_{\omega+1} s'$ iff whenever $s' \xrightarrow{l} t$, $\Phi_{s,t,l}(x)$ is satisfiable in $(L \cup S, L, \rightarrow, s, t, l)$.

⊣

A standard compactness argument (e.g., Sacks [22]) applied to Proposition 1 yields the conclusion that every transition system has an elementary extension over which $\underline{\leftrightarrow} = \underline{\leftrightarrow}_\omega$. Unfortunately, bisimilarity is not preserved under elementary submodels of transition systems (Fernando [9]).

The moral here is put somewhat colorfully in page 96 of Sacks [22] (where to say that a type is 'omitted' is to mean that the type is not satisfiable):

Any fool can realize a type, but it takes a model-theorist to omit one.

To understand this quip, the following sets of first-order formulas are instructive. For simplicity, we work with the case that $|L| = 1$, therefore ignoring labels and taking T instead to be a binary predicate symbol. Let $\chi_{\geq n}(x_1, \ldots, x_n)$ be the formula

$$\bigwedge_{1 \leq i < j \leq n} \quad x_i \neq x_j \ \& \ (\forall y)(T(x_i, y) \ \equiv \ y = x_{i+1})$$

formalizing the picture

$$x_1 \rightarrow x_2 \rightarrow \cdots \rightarrow x_n (\rightarrow ?) \quad ,$$

and let $\varphi_n(x)$ be the formula

$$(\exists x_1) \cdots (\exists x_n) \qquad T(x, x_1) \ \& \ \chi_{\geq n}(x_1, \ldots, x_n) \ \& \ (\forall y) \neg T(x_n, y)$$

saying that x can make a transition to a branch of length n. Next, for every subset A of ω, define the set of formulas

$$\Phi_A(x) \quad = \quad \{\varphi_n(x) \mid n \in A\} \ \cup \ \{\neg \varphi_n(x) \mid n \notin A\} \ .$$

Now, we have two points to make about such sets $\Phi_A(x)$. First, as there are uncountably many subsets A of ω, a model realizing all such sets (which are finitely satisfiable in, for example, the operational semantics specified in §3.1 below) must be uncountable. Thus, not every state in such a model can be denoted by a term (assuming there are only countably many terms). Note, however, that by Proposition 1, only countably many such sets need be realized by a countable transition system for AIP to hold there. Second, if A is infinite and $\Phi_A(x)$ is realized in some model M by a state s such that

$$(M, s) \ \models \ (\forall x) \ (T(\dot{s}, x) \ \supset \ (\forall y)(\forall z)((T(x, y) \ \& \ T(x, z)) \ \supset \ (y = z \ \& \ x \neq y)) \ ,$$

then there is *no* infinite branch from s precisely if the set

$$\{T(\dot{s}, x)\} \quad \cup \quad \{(\exists x_2)(\exists x_3) \cdots (\exists x_n) \ \chi_{\geq n}(x, x_2, x_3, \ldots, x_n) \mid n \geq 2\}$$

is *not* satisfiable (i.e., is *omitted*) in (M, s), even though it is finitely satisfiable there. In short, the Stone space of a theory (in first-order logic or some fragment thereof) may include 'ideal' elements that have no copies in the intended models of the theory. One of the unrealized aims of the present work is to show how well-known recursion-theoretic generalizations of the notion of finiteness in compactness (to the α-finite, due to Kreisel and Barwise) might overcome this defect. The next subsection suggests that this may call for a generalization beyond the paradigmatic setting of hyperarithmetic (i.e., Δ^1_1) sets.

2.2 Logical complexity grounded in mechanical computation

The present subsection strengthens the result in Fernando [9] that $\underline{\leftrightarrow} \notin \Pi^1_1$ for some transition predicate $\succ$ computable in logspace. The next step after first-order logic (styled as Σ^0_1-logic in Girard [11]), Π^1_1-logic includes dynamic logic and various extensions (Harel [14]), where satisfaction over the intended (ω-)models is hyperarithmetic.

Call a transition predicate $\rightarrow$ *strongly extensional* (Aczel [2]) if $\underline{\leftrightarrow}$ is $=$; i.e., $a \underline{\leftrightarrow} b$ implies $a = b$. This notion represents an extreme in the trade-off between the complexity of the transition predicate and the complexity of bisimilarity over that transition predicate.

Theorem 2. *There is a rooted transition system with a transition predicate $\succ$ computable in logspace such that for every rooted transition system with transition predicate $\rightarrow$ bisimilar to it, if $\rightarrow$ is Σ^1_1, then bisimilarity over $\rightarrow$ is not Π^1_1 (whence if $\rightarrow$ is strongly extensional, then $\rightarrow$ is not Σ^1_1).*

Proof. The idea is to work with a primitive recursive order (due to R.O. Gandy) that has an initial segment of height ω_1^{CK} (see, for example, Theorem 5.6.8 in Girard [11], p. 319). That order, call it $\succ$, can be assumed to be computable in logspace, by Grigorieff [12]. So as to apply the notion of a bisimulation on $\succ$, identify $\succ$ with the transition predicate $\{(a, 0, b) \mid a \succ b\}$. Call a $\succ$-state a *non-standard* if there is no hyperarithmetic strictly $\succ$-descending sequence from a, although there is some (non-hyperarithmetic) strictly $\succ$-descending sequence from it. Associate with $\succ$ a root $\hat{r}$ that is non-standard. Such a state exists by the Kleene Basis Theorem (Theorem 5.6.7 in Girard [11], p. 318). Now fix a transition predicate $\rightarrow$ with root r bisimilar to $\hat{r}$ (under $\succ$). For simplicity, let us ignore the label on $\rightarrow$-transitions, and treat $\rightarrow$ as a binary predicate.

> **Lemma.** *For every a such that $r \rightarrow a$, $r \underline{\leftrightarrow} a$ iff there is a strictly $\rightarrow$-descending sequence from a.*
>
> **Proof.** Necessity is clear, since $\rightarrow$ is non-well-founded on r. As for the converse, appeal to the coinductive characterization of $\underline{\leftrightarrow}$ to conclude that every two non-standard states are bisimilar.

Then the restriction of $\rightarrow$ to

$$\{a \mid r \rightarrow a \text{ and } r \not\underline{\leftrightarrow} a\}$$

has height ω_1^{CK}. So by Σ^1_1-boundedness (Lemma 5.6.4 in Girard [11], p. 317), $\rightarrow$ cannot be Σ^1_1 if bisimilarity over it is Π^1_1. $\dashv$

3 Bisimilarity as a congruence

Given a transition system $(L, S, \rightarrow)$ and a set F of functions on S, it may, of course, happen that bisimilarity $\underline{\leftrightarrow}$ is not a congruence with respect to a function f in F; that is, $s_1 \underline{\leftrightarrow} t_1, \ldots, s_n \underline{\leftrightarrow} t_n$ need not imply $f(s_1, \ldots, s_n) \underline{\leftrightarrow} f(t_1, \ldots, t_n)$. One might say that the fault lies with the functions in F, study rules introducing functions relative to which bisimilarity is guaranteed to be a congruence (see, for example, Groote and Vaandrager [13], and the references cited therein), and simply leave the matter at that. Alternatively, the failure of $\underline{\leftrightarrow}$ to be a congruence with respect to functions in F may be construed as a symptom of an incomplete set of transitions (e.g. an inappropriate level of abstraction), suggesting that the transition system be modified so as to turn bisimilarity into a congruence. A natural question is whether this can be carried out effectively — i.e., within a universal operational semantics, the nature of which we first make precise.

3.1 A universal operational semantics

Fix a binary primitive recursive function $\langle \cdot, \cdot \rangle : \omega \times \omega \rightarrow \omega$ and unary primitive recursive functions $i, j : \omega \rightarrow \omega$ such that for all $n, m < \omega$,

$$i(\langle n, m \rangle) = n \ , \quad j(\langle n, m \rangle) = m \quad \text{and} \quad \langle i(n), j(n) \rangle = n \ .$$

Let U be an r.e. set such that $\{U_e\}_{e<\omega}$ enumerates all r.e. sets, where

$$U_e \quad = \quad \{n \mid \langle e, n\rangle \in U\} \ ,$$

and define

$$\rightsquigarrow \quad = \quad \{(s, l, s') \mid i(s) = i(s') \text{ and } \langle j(s), \langle l, j(s')\rangle\rangle \in U_{i(s)}\} \ .$$

Note that $\rightsquigarrow$ is slightly different from a standard parametrization of binary r.e. predicates (although the modifications are minor enough). It is, in a suitable sense, an effective disjoint sum of all r.e. transition systems.

Proposition 3. *The predicate $\rightsquigarrow$ is r.e. and contains a copy of every r.e. transition predicate in that for every r.e. $\rightarrow$, there is a function f (that is, in fact, 1-1 and recursive) with domain equal to the set of $\rightarrow$-states (i.e., $\{s \mid (\exists l)(\exists t)\ s \xrightarrow{l} t \text{ or } t \xrightarrow{l} s\}$) such that*

$$s \xrightarrow{l} t \quad \textit{iff} \quad f(s) \overset{l}{\rightsquigarrow} f(t) \ ,$$

and, moreover,

$$f(s) \overset{l}{\rightsquigarrow} u \ \textit{ or } \ u \overset{l}{\rightsquigarrow} f(s) \quad \textit{implies} \quad (\exists t)\ f(t) = u \ .$$

Proof. Given an r.e. $\rightarrow$, define f on $\rightarrow$-states s by $f(s) = \langle e, s\rangle$ where $U_e = \{\langle s, \langle l, t\rangle\rangle \mid s \xrightarrow{l} t\}$. $\dashv$

Inasmuch as the transitions of a machine must be generated mechanically, $\rightsquigarrow$ comprises (by Church's thesis) a universal mechanical transition predicate. Observe that, contrary to algebraic term models, $\rightsquigarrow$ is defined without any $n+1$-ary function symbols. Hence, bisimilarity over the transition system $(\omega, \omega, \rightsquigarrow)$ is vacuously a congruence. But what about expanding an $\{\dot{L}, T\}$-model to interpret function symbols?

3.2 Turning bisimilarity into a congruence

Fix a transition system $(L, S, \rightarrow)$ and a set of function symbols to be interpreted over S according to certain transition rules. For example, F may consist of unary function symbols f and is_f, and a 0-ary function symbol (i.e., constant) 0 subject to the rules

$$\frac{p \xrightarrow{l} q}{f(p) \xrightarrow{l} f(q)} \qquad \frac{}{is_f(f(p)) \xrightarrow{a} p} \ ,$$

meaning that 0 can make no transition, and that for all p, p and $f(p)$ must make the same transitions, and $is_f(p)$ can make a transition precisely if there is a q such that $f(q) = p$, in which case $is_f(p) \xrightarrow{a} q$. Observe that $\leftrightarrows$ is not a congruence with respect to is_f since $0 \leftrightarrows f(0)$ but $is_f(0) \not\leftrightarrows is_f(f(0))$, assuming (as is standard paractice) an 'initial' interpretation where 0 is not in the image of f. Now, without altering $\rightarrow$ on S and L, we can add the label is_f and the rule

$$\frac{}{f(p) \xrightarrow{is_f} f(p)} \ ,$$

thereby negating $0 \leftrightarrows f(0)$. (One can make do with a single additional label, if new states can be introduced; see, for example, the discussion of labelled systems in Aczel [2].)

Another example (from Vaandrager [23]) is the rule

$$\frac{}{p + p \xrightarrow{b} 0} \ ,$$

yielding $0 \leftrightarrows 0 \| 0$ but $0 + 0 \not\leftrightarrows 0 + 0 \| 0$. We can negate $0 \leftrightarrows 0 \| 0$ by adding the label is_0, and the transition $0 \xrightarrow{is_0} 0$ (which should be distinguished from other transitions to preserve the intuition that 0 represents

inaction). As this practice of introducing (syntactic) labels can, however, be abused, we can (alternatively) affirm $0+0 \underline{\leftrightarrow} 0+0\|0$ (noting that transitions can be introduced to increase $\underline{\leftrightarrow}$, as well as to decrease $\underline{\leftrightarrow}$). Without disturbing L or S, complete $\rightarrow$ to $\rightarrow_F$ by adding (to $\rightarrow$) the rule

$$\frac{f(s_1,\ldots,s_n) \xrightarrow{l} t \qquad s_1 \underline{\leftrightarrow} s_1' \qquad \cdots \qquad s_n \underline{\leftrightarrow} s_n'}{f(s_1',\ldots,s_n') \xrightarrow{l}_F t}$$

for every n-ary $f \in F$, where $\underline{\leftrightarrow}$ is bisimilarity over $\rightarrow$. That is, define

$$s \xrightarrow{l}_F t \quad \text{iff} \quad s \xrightarrow{l} t \text{ or } (\exists n)(\exists n\text{-ary } f \in F)(\exists s_1,\ldots,s_n,s_1',\ldots,s_n')$$
$$s = f(s_1',\ldots,s_n') \;\&\; s_1 \underline{\leftrightarrow} s_1' \;\&\; \cdots \;\&\; s_n \underline{\leftrightarrow} s_n' \;\&\; f(s_1,\ldots,s_n) \xrightarrow{l} t\ .$$

Proposition 4. *Bisimilarity $\underline{\leftrightarrow}_F$ over $\rightarrow_F$ is congruence with respect to every function in F. Furthermore, $\underline{\leftrightarrow}_F = \underline{\leftrightarrow}$ iff $\underline{\leftrightarrow}$ is a congruence with respect to every function in F.*

The proof is immediate.

In view of the tremendous complexity $\underline{\leftrightarrow}$ may have (Theorem 2), the question arises as to whether given an r.e. $\rightarrow$, the predicate $\rightarrow_F$ has a copy within $\rightsquigarrow/\underline{\leftrightarrow}$, where

$$s \xrightarrow{l}/\underline{\leftrightarrow} t \quad \text{iff} \quad (\exists s',t')\ s \underline{\leftrightarrow} s' \;\&\; t \underline{\leftrightarrow} t' \;\&\; s' \xrightarrow{l} t'\ .$$

(Note that $\rightsquigarrow/\underline{\leftrightarrow}$ is $\rightsquigarrow_F$ where F consists of a single unary function symbol interpreted as the identity function.) Just what can be accomodated within $\rightsquigarrow/\underline{\leftrightarrow}$ is investigated (with partial results) next.

4 Reductions between transition predicates

Bisimilarity is sometimes said to be the finest of various notions of equivalence imposed on transition systems because of the special kind of transition systems on which it is defined — such transition systems typically enjoying the following characteristics: (i) the transition relation is finitely branching, and (ii) bisimilarity between so-called finite projections is decidable. Without spelling out exactly what (i) and (ii) mean, suffice it to say that together, (i) and (ii) constrain bisimilarity to be Π_1^0 (e.g., Bloom, Istrail and Meyer [5]). But then the complexity (and hence, the scope) of bisimilarity over arbitrary r.e. transition relations can well exceed Π_1^0 (as established in Theorem 2, for a transition predicate computable in logspace!). (The necessity of infinite branching for modelling features of computation met in practice, and for a universal operational semantics is discussed in Vaandrager [23], which then focuses, all the same, on finite branching.) The present section measures the complexity of $\rightsquigarrow/\underline{\leftrightarrow}$, and then turns to 'abstract' notions of transitions.

4.1 Many-one reductions

Recall that a set A is *many-one reducible to* a set B if there is a total recursive function f such that

$$a \in A \quad \text{iff} \quad f(a) \in B\ .$$

Theorem 5. *$\rightsquigarrow/\underline{\leftrightarrow}$ is Σ_1^1-complete (under many-one reductions).*

Proof (sketch). $\rightsquigarrow/\underline{\leftrightarrow}$ is manifestly Σ_1^1. For Σ_1^1-hardness, let $\rightarrow$ be the usual r.e. order underlying Church-Kleene notations $\mathcal{O}$, and define inessential variants of $\mathcal{O}$ and Feferman-Spector notations $\mathcal{O}^*$ by

$$\begin{aligned} \mathcal{O}_0 &= \{a \mid \text{there is no infinite } \rightarrow\text{-descending sequence from } a\} \\ \mathcal{O}_0^* &= \{a \mid \text{there is no hyperarithmetic infinite } \rightarrow\text{-descending sequence from } a\} \end{aligned}$$

respectively. (The inessential difference here is that for $b \in \mathcal{O}$, if $b \rightarrow 3 \cdot 5^e$, then $3 \cdot 5^e$ must name a limit ordinal; and similarly for $\mathcal{O}^*$.) Fix an $a \in \mathcal{O}_0^* - \mathcal{O}_0$, and note that $\mathcal{O}_0^*$ is $\{b \in \omega \mid a \rightarrow/\underline{\leftrightarrow} b\}$. It suffices to show that the argument for the Σ_1^1-hardness of $\mathcal{O}^*$ established by Feferman [7] can withstand the

slight modification to $\mathcal{O}_0^*$ — i.e., that $\{b \in \omega \mid a \to /\underline{\leftrightarrow}\ b\}$ is Σ_1^1-hard. Proceed as follows. Appeal to the Spector-Gandy Theorem (e.g., Moschovakis [19], p. 117) to characterize a Σ_1^1-set A by

$$x \in A \quad \text{iff} \quad (\forall_{hyp} g)(\exists y) R(\overline{g}(y), x)$$

(where $\forall_{hyp}$ is a universal quantifer ranging over hyperarithmetic functions, R is recursive, and $\overline{g}(y) = \langle g(0), \ldots, g(y)\rangle$), and show that Kleene's many-one reduction of the Π_1^1 predicate $(\forall g)(\exists y)R(\overline{g}(y), x)$ to $\mathcal{O}$ also reduces $(\forall_{hyp} g)(\exists y)R(\overline{g}(y), x)$ to $\mathcal{O}_0^*$.

4.2 Abstract transitions

To see that bisimilarity subsumes input/output equivalence, we prove an embarrassingly simple proposition.

Proposition 6. *Let P be some set of natural numbers (e.g., syntactic terms), and $[\![\cdot]\!]$ some function with domain P. Assume that there is a map $[\cdot] : P \to \mathrm{Power}(\omega)$ such that (i) $[\![p]\!] = [\![p']\!]$ iff $[p] = [p']$, and (ii) the predicate $n \in [p]$ is r.e. (in n and p). Then there is an r.e. transition predicate $\to$ such that for all $p, p' \in P$,*

$$[\![p]\!] = [\![p']\!] \quad \textit{iff} \quad p \underline{\leftrightarrow} p' \ .$$

Proof. Defining $\to$ to be $\{(p, n, p) \mid n \in [p]\}$ will do. The burden need not be shoved entirely on the labels, as $\to$ may just as easily be defined on a singleton label set by adding fresh states (again, see, for example, the discussion of labelling in Aczel [2]), although the equivalence must then be weakened say to $[\![p]\!] = [\![p']\!]$ iff $\langle p, 0\rangle \underline{\leftrightarrow} \langle p', 0\rangle$. $\dashv$

Proposition 6 applies when the notion of process equivalence is axiomatizable (i.e., $[\![p]\!] = [\![p']\!]$ is r.e. in p, p'), for then we can define $[\cdot]$ by $[p] = \{p' \mid [\![p]\!] = [\![p']\!]\}$. But Proposition 6 may also apply when $[\![p]\!] = [\![p']\!]$ is not r.e., as in the case of input/output equivalence, which is Π_2^0-complete. Indeed, it is a small (and not unnatural) step from the transitions

$$p \overset{d,d'}{\to} p'$$

of, for instance, De Boer, Kok, Palamidessi and Rutten [6] (where the data pair label d, d' replaces the more traditional atomic action a that presumably takes d to d') to a composite version p_* of p closed under sequential composition of transitions

$$\frac{p \overset{d,d'}{\to} p'}{p_* \overset{d,d'}{\to} p'} \qquad \frac{p_* \overset{d,d'}{\to} p' \qquad p' \overset{d',d''}{\to} p''}{p_* \overset{d,d''}{\to} p''} \ ,$$

and then to an "atomized" form p_{at} of p given by

$$\frac{p_* \overset{d,d'}{\to} \surd}{p_{at} \overset{d,d'}{\to} \surd}$$

where all transitions to intermediate states are abstracted away, $\surd$ being a terminal state marking successful termination. Thus, input/output equivalence can be recovered by defining

$$[p] \ = \ \{(d, d') \mid p_{at} \overset{d,d'}{\to} \surd\} \ .$$

(A similar reduction works for trace equivalence.)

We close this section with two somewhat disparate remarks. First, rather than formulating the reductions as functions on states (interpreted say, in $\rightsquigarrow$), the functions can be introduced on transition systems (with copies in $(\omega, \omega, \rightsquigarrow)$). And second, although input/output equivalence is Π_2^0-complete, and $\rightsquigarrow /\underline{\leftrightarrow}$ is Σ_1^1-complete, it turns out that there is a Π_2^0-transition predicate (with a Δ_3^0 strongly extensional form) without a ('trace-equivalent') copy in $\rightsquigarrow /\underline{\leftrightarrow}$ (Fernando [10]).

5 Bisimulations on hereditarily effective operations

The present section is devoted to analyzing input/output statements $\{e\}(x) \simeq y$ (over some fixed standard enumeration of the partial recursive functions $\{e\}$, $e \geq 0$) as a special case of (terminating, possibly composite) transitions

$$\{e\}(x) \simeq y \quad \text{iff} \quad e \stackrel{x,y}{\rightarrow} \surd . \tag{1}$$

Towards that end, it will prove useful to relax the rigidity of labels in the definition of a bisimulation, or equivalently, to redefine the transition predicate slightly. Also, bisimulations that are not necessarily maximal will come in handy.

5.1 A concrete presentation of partial equivalence relations

The system of *pure types* τ defined by

$$\tau \quad ::== \quad 0 \mid \tau \Rightarrow 0$$

admits a straightforward interpretation by the *hereditarily recursive operations* $\{HRO_\tau\}_\tau$

$$\begin{aligned} HRO_0 &= \omega \\ HRO_{\tau\Rightarrow 0} &= \{e \mid (\forall x \in HRO_\tau)\ \{e\}(x)\downarrow\} . \end{aligned}$$

An interpretation with a more extensional character can be given by defining an "equality" relation alongside it, as in the *hereditarily effective operations* $\{(HEO_\tau, =_\tau)\}_\tau$

$$\begin{aligned} HEO_0 &= \omega \\ =_0 &= \{(n,n) \mid n \in \omega\} \\ HEO_{\tau\Rightarrow 0} &= \{e \mid (\forall x, y \text{ such that } x =_\tau y)\ \{e\}(x)\downarrow ,\ \{e\}(y)\downarrow \text{ and } \{e\}(x) \simeq \{e\}(y)\} \\ =_{\tau\Rightarrow 0} &= \{(x,y) \mid x \in HEO_{\tau\Rightarrow 0} ,\ y \in HEO_{\tau\Rightarrow 0} \text{ and } (\forall z \in HEO_\tau)\ \{x\}(z) \simeq \{y\}(z)\} . \end{aligned}$$

(Note that the "equality" relation $=_{0\Rightarrow 0}$ is already Π^0_2-complete, and the complexity of equality only grows worse as the types become more complicated.) This interpretation (due to Kreisel [16]) has since been generalized into so-called partial equivalence relations (see, for example, Mitchell [18]) and applied to various type systems (such as Girard's system F). A *partial equivalence relation* (per) is a binary relation $R \subseteq \omega \times \omega$ that is symmetric and transitive. Following $=_{\tau\Rightarrow 0}$, define the *exponentiation* $R \Rightarrow S$ of pers R and S by

$$R \Rightarrow S \quad = \quad \{(e,e') \mid (\forall n, m \text{ such that } nRm)\ \{e\}(n)\downarrow ,\ \{e'\}(m)\downarrow \text{ and } \{e\}(n)S\{e'\}(m)\} .$$

To define products on pers, it will be useful for (1) to adopt the following coding conventions.

Coding conventions. Assume that 0 codes the totally undefined empty function (i.e., $\{0\} = \emptyset$), and that $\langle 0,0\rangle = 0$, for some pairing system $\langle\cdot,\cdot\rangle, i, j$, as described in §3.1. Let $[\cdot,\cdot]$ be a 1-1 binary recursive function such that

$$\{[s,s']\}(x) \quad = \quad \langle \{s\}(i(x)), \{s'\}(j(x))\rangle$$

and $[0,0] = 0$. A moment's thought on how to construct $[\cdot,\cdot]$ (over an enumeration given say, by Turing machines) should be enough to see that $[\cdot,\cdot]$ can be assumed to have recursive projections. Now, define the *product* $R \times S$ of pers R, S by

$$R \times S \quad = \quad \{([n,m],[n',m']) \mid nRn' \text{ and } mSm'\} .$$

Note that the careful attention to coding goes against the usual category-theoretic grain, inasmuch as that is insensitive to such syntactic details (on which, however, (1) is based).

5.2 Turning a per into a bisimulation

In turning a per into a bisimulation, an immediate difficulty with transitions given by (1) is that a per whose domain includes e may fail to be a bisimulation relative to (1). The problem is that labels may be identified, in that, for example, the definition of a per $R \Rightarrow S$ allows $e(R \Rightarrow S)e'$, xRx and $\{e\}(x) \not\simeq \{e'\}(x)$ so long as $\{e\}(x)S\{e'\}(x)$. Accordingly, it is useful to modify the transition predicate slightly by closing it under some equivalence $\sim$ on labels. More precisely, a binary relation $R \subseteq S \times S$ is a *$\sim$-bisimulation* on a transition predicate $\rightarrow$ if R is a bisimulation on the transition system with label set $L = \{l \mid (\exists s, t \in S)\ s \xrightarrow{l} t\}$, and with transition predicate

$$\{(s, l, t) \in S \times L \times S \mid (\exists m \sim l)\ s \xrightarrow{m} t\}.$$

Now comes our recipe for cooking up $\rightarrow$ and $\sim$, given a per $\approx$. Roughly, the idea is to decompose a per $\approx$ into pers $\sim_d$ and $\sim_r$ such that $\sim_d \Rightarrow \sim_r$ approximates $\approx$, and then to set $\sim$ essentially to $\sim_d \times \sim_r$. More precisely, given a per $\approx$, define

$$\sim \quad = \quad \{((x, y), (x', y')) \mid x \sim_d x' \text{ and } y \sim_r y'\}$$

where $\sim_d$ and $\sim_r$ are defined simultaneously as the ($\subseteq$-)largest solution to

$$\sim_d \quad = \quad \{(x, x') \mid (\forall e, e' \text{ such that } e \approx e')\ \{e\}(x)\downarrow\ ,\ \{e'\}(x')\downarrow \text{ and } \{e\}(x) \sim_r \{e'\}(x')\}$$

and

$$\sim_r \quad = \quad \{(y, y') \mid (\exists e, e', x, x')\ \ e \approx e'\ ,\ x \sim_d x'\ ,\ \{e\}(x) \simeq y \text{ and } \{e'\}(x') \simeq y'\}\ .$$

That is, $\sim_d$ and $\sim_r$ are calculated coinductively as $\bigcap_\alpha \sim_d^\alpha$ and $\bigcap_\alpha \sim_r^\alpha$, respectively, where the approximations $\sim_d^\alpha$ and $\sim_r^\alpha$, indexed by ordinals α, are given by $\sim_d^0 = \omega \times \omega = \sim_r^0$,

$$\begin{aligned} \sim_d^{\alpha+1} \quad &= \quad \{(x, x') \mid (\forall e, e' \text{ such that } e \approx e')\ \{e\}(x)\downarrow\ ,\ \{e'\}(x')\downarrow \text{ and } \{e\}(x) \sim_r^\alpha \{e'\}(x')\} \\ \sim_r^{\alpha+1} \quad &= \quad \{(y, y') \mid (\exists e, e', x, x')\ \ e \approx e'\ ,\ x \sim_d^\alpha x'\ ,\ \{e\}(x) \simeq y \text{ and } \{e'\}(x') \simeq y'\} \end{aligned}\ .$$

and for limit λ, $\sim_d^\lambda = \bigcap_{\alpha<\lambda} \sim_d^\alpha$, and $\sim_r^\lambda = \bigcap_{\alpha<\lambda} \sim_r^\alpha$. Finally, identifying $\surd$ with 0 (a $\{\cdot\}$-index for the totally undefined function), define the transition predicate $\rightarrow \subseteq dom(\approx) \times dom(\sim) \times \{\surd\}$ by restricting (1) to $e \in dom(\approx)$ and $x \in dom(\sim_d)$

$$\rightarrow \quad = \quad \{(e, (x, y), \surd) \mid e \in dom(\approx),\ x \in dom(\sim_d) \text{ and } \{e\}(x) \simeq y\}\ .$$

Proposition 7.

(i) $\approx \cup\ \{(\surd, \surd)\}$ *is a* $\sim$*-bisimulation on* $\rightarrow$.

(ii) If $\approx$ *is the exponentiation* $R \Rightarrow S$ *of pers* R *and* S, *then* $\sim_d \supseteq R$ *and* $\sim_r \supseteq S$.

(iii) If $\approx$ *is the product* $R \times S$ *of pers* R *and* S, *then*

$$\begin{aligned} \sim_d \quad &= \quad \{(\langle n, m\rangle, \langle n', m'\rangle) \mid n \sim_d^R n' \text{ and } m \sim_d^S m'\} \\ \sim_r \quad &= \quad \{(\langle n, m\rangle, \langle n', m'\rangle) \mid n \sim_r^R n' \text{ and } m \sim_r^S m'\}\ . \end{aligned}$$

Verifying Proposition 7 consists of routine calculations (the real work having been the formulation of the proposition).

Observe that two different pers can yield the same transition relation. Thus, $\approx \cup\ \{(\surd, \surd)\}$ is not necessarily the largest $\sim$-bisimulation on $\rightarrow$ (or $\rightarrow \cup\ \{(\surd, *, \surd)\}$ for some fresh label $*$). Also, for pers $\approx$ generated from $\{(n, n) \mid n \in \omega\}$ by $\Rightarrow$, $\times$, and possibly other type constructs, $\sim_r$ may be expected to be given explicitly by

$$\sim_r \quad = \quad \{(y, y') \mid (\exists a, b, x)\ \ a \approx b\ ,\ \{a\}(x) \simeq y \text{ and } \{b\}(x) \simeq y'\}\ ,$$

thereby avoiding the coinductive complications above.

The construction of $\sim$ and $\rightarrow$ from $\approx$ above yields a special instance $F(\approx)$ of structures $(=_L, =_S, \Longrightarrow)$ where $\Longrightarrow$ is a transition predicate with label equivalence $=_L$ and state equivalence $=_S$. Take

$$F(\approx) \;=\; (\sim,\ \approx \cup \{(\surd,\surd)\},\ \{(s,l,t) \in dom(\approx) \times L \times \{\surd\} \mid (\exists m \sim l)\ s \xrightarrow{m} t\})\ .$$

Preservation of $\times$ and $\Rightarrow$ under F

$$\begin{aligned} F(\approx \times \approx') &= F(\approx) \times F(\approx') \\ F(\approx \Rightarrow \approx') &= F(\approx) \Rightarrow F(\approx') \end{aligned}$$

can be established provided an arrow from $(=_L, =_S, \Longrightarrow)$ to $(='_L, ='_S, \Longrightarrow')$ is given by a pair (e_L, e_S) such that (the e_Lth partial recursive function) $\{e_L\}$ is defined on $dom(=_L)$, while $\{e_S\}$ is defined on $dom(=_S)$, and for all $l, m \in dom(=_L)$ and $s, t \in dom(=_S)$,

$$\begin{aligned} l =_L m \quad &\text{implies} \quad \{e_L\}(l) ='_L \{e_L\}(m) \\ s =_S t \quad &\text{implies} \quad \{e_S\}(s) ='_S \{e_S\}(t) \end{aligned}$$

and

$$s \overset{l}{\Longrightarrow} t \quad \text{implies} \quad \{e_S\}(s) \overset{\{e_L\}(l)}{\Longrightarrow'} \{e_S\}(t)\ .$$

Admittedly, $F(\approx)$ represents a very coarse level of abstraction, all of its transitions ending at the terminal state $\surd$. On the other hand, such instances are natural starting points for generalizing input/output statements to transitions into and from intermediate computational states.

6 Discussion

To properly appreciate the manner in which bisimulations generalize input/output equivalence, the data on which the notion of input/output is defined must be brought out (rather than, as is common practice, abstracted away in atomic actions). The dependence of transitions on data is studied in Fernando [9]. We have concentrated here on using the logical complexity of bisimulations to relate different levels of abstraction, implicitly disputing the claim that bisimilarity is an inappropriate notion of program equivalence because it is too fine. Another limitation often associated with bisimulations is the view that the transition system conception of computation is based on graphs over which

(i) paths are traversed by sequential composition, and

(ii) only disjunctive (and not conjunctive) branching is supported

(thereby reducing parallelism to sequential composition and non-deterministic choice). This is particularly striking in the polynomials of Baeten and Weijland [3], where multiplication is sequential composition, and addition is non-deterministic choice. But it is simple enough to formulate conjunctive branching by transitions between sets of states (e.g., Peleg [21], Fernando [8]). And there is no necessity in assuming labels refer to 'atomic' transitions that can be freely composed sequentially (nor in resorting to so-called τ-steps to introduce compound transitions). Allowing for more complex transitions, the intuition behind an nth projection loses much of its force — a loss of some consequence, given that the Approximation Induction Principle is defined in terms of these projections. Indeed, the highly uncountable size (Bergstra and Klop [4]) of so-called projective limit and complete metric space models (which, by construction, satisfy AIP) belie the practice of applying AIP to simple (e.g., image finite) transition systems. Setting aside such coarse cardinality considerations, Proposition 1 above shows how AIP is validated by indiscriminately realizing (not necessarily uncountably many) finitely satisfiable sets of formulas (e.g., Abramsky [1]). But to faithfully model what can be computed mechanically, one must pay attention to omitting some finitely satisfiable sets as well. Omitting types arguments commonly suggest moving from first-order logic to ω-logic (the ω-completeness theorem being an important corollary of the omitting types theorem) — or, in the 'logical complexity'

terminology of Girard [11], stepping up from Σ^0_1-logic to Π^1_1-logic. The rough idea is that while Σ^0_1-logic may suffice for a syntactic analysis of mechanical transitions (inasmuch as these transitions form, according to Church's thesis, a Σ^0_1-set relative to some numerical coding), a semantic analysis inevitably gives rise to more complex (set-theoretic) concepts. In fact, the semantic abstractions that the notion of a bisimulation introduces easily fall beyond Π^1_1-logic. This explosion in logical complexity is related in Fernando [10] to generalized recursion, and, in particular, to the next admissible set (e.g., Moschovakis [19]).

References

[1] Samson Abramsky. A domain equation for bisimulation. *Information and Computation*, 92, 1991.

[2] Peter Aczel. *Non-Well-Founded Sets.* CSLI Lecture Notes Number 14, Stanford, 1988.

[3] J.C.M. Baeten and W.P. Weijland. *Process Algebra.* Cambridge Tracts in Theoretical Computer Science 18. Cambridge University Press, 1990.

[4] J. Bergstra and J.W. Klop. A convergence theorem in process algebra. Technical Report CS-R8733, Centre for Mathematics and Computer Science (CWI), Amsterdam, July 1987.

[5] B. Bloom, S. Istrail, and A.R. Meyer. Bisimulation can't be traced. In *Proc. 15th ACM Symp. on Principles of Programming Languages*, 1988. To appear in *J. Assoc. Computing Machinery.*

[6] F.S. de Boer, J.N. Kok, C. Palamidessi, and J.J.M.M. Rutten. The failure of failures in a paradigm for asynchronous communication. In J.C.M. Baeten and J.F. Groote, editors, *Proc. Concur '91*, LNCS 527. Springer-Verlag, Berlin, 1991.

[7] Solomon Feferman. Constructive pseudo-well-orderings. *Notices Amer. Math. Soc.*, 9:136, 1962.

[8] Tim Fernando. A higher-order extension of constraint programming in discourse analysis. Position paper for the First Workshop on Principles and Practice of Constraint Programming, Rhode Island, April 1993.

[9] Tim Fernando. Bisimulations and predicate logic. Technical Report CS-R9333, Centre for Mathematics and Computer Science (CWI), Amsterdam, May 1993. To be presented in Computer Science Logic '93, Swansea, September 1993.

[10] Tim Fernando. A path from generalized recursion back to mechanical computation. In preparation (to be presented in a conference on 'Proofs and Computation', Oslo, September 1993).

[11] Jean-Yves Girard. *Proof Theory and Logical Complexity*, volume 1. Bibliopolis, Napoli, 1987.

[12] Serge Grigorieff. Every recursive linear ordering has a copy in DTIME-SPACE($n, log(n)$). *Journal of Symbolic Logic*, 55(1), 1990.

[13] J.F. Groote and F.W. Vaandrager. Structured operational semantics and bisimulation as a congruence. *Information and Computation*, 100(2), October 1992.

[14] David Harel. Dynamic logic. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume 2. D. Reidel, 1984.

[15] M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *J. Assoc. Computing Machinery*, 32(1), 1985.

[16] Georg Kreisel. Interpretation of analysis by means of constructive functionals of finite types. In A. Heyting, editor, *Constructivity in Mathematics.* North-Holland, Amsterdam, 1959.

[17] Robin Milner. *Communication and Concurrency.* Prentice Hall, 1989.

[18] John C. Mitchell. Type systems for programming languages. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science: Formal Models and Semantics*, volume B. Elsevier, Amsterdam, 1990.

[19] Yiannis N. Moschovakis. *Elementary Induction on Abstract Structures.* North-Holland, Amsterdam, 1974.

[20] David Park. Concurrency and automata on infinite sequences. In P. Deussen, editor, *Proc. 5th GI Conference*, LNCS 104. Springer-Verlag, Berlin, 1981.

[21] David Peleg. Concurrent dynamic logic. *J. Assoc. Computing Machinery*, 34(2), 1987.

[22] Gerald E. Sacks. *Saturated Model Theory.* W.A. Benjamin, Reading, Mass., 1972.

[23] Frits W. Vaandrager. Expressiveness results for process algebras. In J.W. de Bakker et al., editor, *Proc. REX Workshop*, LNCS. Springer-Verlag, Berlin, 1993. Also appears as CWI Technical Report CS-R9301, Amsterdam, January 1993.