# Bisimulations and predicate logic

R.T.P. Fernando

Computer Science/Department of Software Technology

**Report CS-R9333 May 1993**

# Bisimulations and Predicate Logic

Tim Fernando
fernando@cwi.nl

*CWI, P.O. Box 4079, 1009 AB Amsterdam, The Netherlands*

**Abstract**

Elementary (first-order) and non-elementary (set-theoretic) aspects of the largest bisimulation are considered, with a view towards analyzing operational semantics from the perspective of predicate logic. The notion of a bisimulation is employed in two distinct ways: (i) as an extensional notion of equivalence on programs (or processes) generalizing input/output equivalence (at a cost exceeding $\Pi^1_1$ over certain transition predicates computable in *log* space), and (ii) as a tool for analyzing the dependence of transitions on data (which can be shown to be elementary or non-elementary, depending on the formulation of the transitions). These ways contrast not only in the logical dependence they uncover, but also in their use of sets as abstractions describing mechanical computation.

*1991 Mathematics Subject Classification*: 03B70.
*1991 Computing Reviews Categories*: D.3.1.
*Keywords and Phrases*: bisimulation, transition system, operational semantics, predicate logic, back-and-forth, (co-)induction, compactness, (non-standard) ordinal notations, generalized extensionality.
*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').
My thanks to S. Feferman for many kindnesses through long and short distances, to G. Jäger for a fruitful month-long visit to Berne, to J.W. Klop for helpful meetings at CWI, and to Ph. Darondeau for useful remarks, including corrections.

Bisimulations (Park [31]) provide a notion of equivalence on states undergoing transitions. This equivalence, called bisimilarity and denoted $\underline{\leftrightarrow}$, has proved to be of interest both to theoretical investigations into the semantics of programs, and to more practical work directed towards the automatic verification of certain specifications. In employing bisimilarity as a computational tool, one is understandably concerned that bisimilarity fall within the realm of mechanical decidability, isolating, if necessary, conditions (on transitions) pushing down its complexity (see Christensen, Hirshfeld and Moller [14] and the references cited therein). From a theoretical standpoint, however, it makes sense to analyze the notion of a bisimulation in its fullest generality and glory, keeping in mind that the greater the scope of a notion, the more potentially interesting it is as an object of study. In particular, given the proliferation of various notions of equivalence on programs, the question arises as to whether these notions can be reduced to bisimilarity under suitable translations of the underlying transition systems (the intuition being that a transition system represents a fixed level of abstraction). Insofar as the logical complexity of bisimilarity is measured by the existence of some such translations, there is interest from the theoretical side in investigating the (full) logical complexity of bisimilarity (however astronomical that may be, relative to mechanical computation) and not only, as already mentioned, in introducing assumptions that lower its complexity. A shift in perspective may be necessary for some, but the perspective that is advocated belongs, in fact, to a well-established logical tradition — namely, predicate logic and generalized notions of computation (exceeding the reach of machines) developed to analyze it.

A consideration crucial to such an analysis is the question of approximating bisimilarity finitarily. Such approximations are usually given according to the co-inductive construction of bisimilarity (e.g., Milner [30]), the finitary fragment $\underline{\leftrightarrow}_\omega$ of which is called observational equivalence in Hennessy and Milner [26]. (Precise definitions are reviewed in section 1 below.) A (modal) logical characterization of $\underline{\leftrightarrow}_\omega$ is provided in Hennessy

Report CS-R9333
ISSN 0169-118X
CWI
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

and Milner [26], which is developed further in Abramsky [1] to construct processes topologically. It is well-known, however, that bisimilarity and $\underline{\leftrightarrow}_\omega$ do not coincide over the simplest cases of infinite branching, in response to which, the modal language might be closed under infinitary disjunction and conjunction. But a free-wheeling appeal to such infinitary constructs begs the problem of analyzing the effectiveness of the notion of infinity introduced, and also poses a problem for the machinery of Stone duality employed in Abramsky [1] (spoiling, as it does, the compactness of the logic and topological space derived from it). Avoiding any connective whatsoever, one might opt (as in the textbook Baeten and Weijland [5]) for an equational logic with an (infinitary) inference rule called the Approximation Induction Principle (AIP), carrying, in cases where bisimilarity is not r.e., a good deal of the burden of logical completeness (e.g. Aceto, Bloom and Vaandrager [2]). But under an interpretation of equality as bisimilarity, AIP simply formalizes the assertion that bisimilarity is $\underline{\leftrightarrow}_\omega$ (which, as already noted, fails when branching can be infinite), and is therefore not a sound rule. The relationship between bisimilarity and $\underline{\leftrightarrow}_\omega$ is analyzed below in terms of compactness, an essential feature of which, in the setting of predicate logic (in contrast to the modal propositional logics of Hennessy and Milner [26] and Abramsky [1]) is the generation of "non-standard" models (à la Abraham Robinson). This point is brought out concretely by Theorem A$'$ (in section 2), which establishes the non-elementary character of bisimilarity via a first-order compactness argument. The author suspects that there is more to be mined in compactness, especially in its generalized form involving admissible sets (Barwise [7]). Making this suspicion plausible is one of the aims of the present paper, which proceeds as follows.

After reviewing some preliminary definitions and facts in section 1, a couple of basic results are presented in section 2 concerning (respectively) the co-inductive characterization of bisimilarity, and the back-and-forth nature of the operator defining the notion of a bisimulation. The first result is the abovementioned Theorem A$'$, while the second (Theorem B$'$) concerns certain transitions defined from data models, and analyzes how these transitions depend on data by turning a bisimulation (back-and-forth) into an isomorphism between the data models. The data dependence of transitions is studied further in section 3, where it is shown (through an omitting types argument) to be first-order (Theorem 4, Corollary 5), by internalizing non-determinism in states given as sets (following the well-known construction of a deterministic finite automaton from a non-deterministic one). Section 4 studies bisimilarity as an extensional notion of equivalence on programs, generalizing input/output equivalence (a $\Pi^0_2$-notion) into an equivalence that may fall outside of $\Pi^1_1$ (Theorem 9, Corollary 10). This explosion in logical complexity is a measure of the scope of bisimilarity resting heavily on infinite branching. Accordingly, some motivation for considering infinite non-determinism might be in order. Beyond the mere fact that an r.e. transition predicate can support infinite branching (which, afterall, is the natural limit of unbounded finite branching), there is the point made (for example) in Vaandrager [33] that "if the machines ... are not in control of all their transitions, then one can argue that ... the requirement of finite branching is too restrictive" to analyze, for instance, inputs and random assignments.

## 1 Fundamental definitions and facts

The present section reviews some well-known material, found, for example, in Milner [30]. A *(labelled) transition system* is a triple $\langle L, S, \rightarrow\rangle$ where $\rightarrow\ \subseteq S \times L \times S$ (and the transition $(s, l, s') \in\ \rightarrow$ is written $s \xrightarrow{l} s'$). Given transition systems $\langle L, S, \rightarrow\rangle$ and $\langle L, S', \rightarrow'\rangle$ over the same label set $L$, a relation $R \subseteq S \times S'$ is a *bisimulation* if whenever $sRs'$ then for all $l \in L$,

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

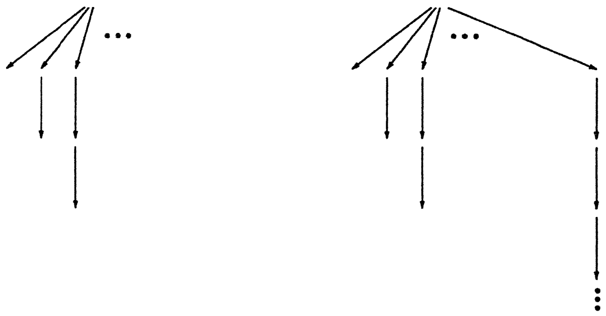
States $s$ and $s'$ are said to be *bisimilar*, which we write as $s \underline{\leftrightarrow} s'$, if there is a bisimulation relating $s$ to $s'$. Note that the relation $\underline{\leftrightarrow}$ of *bisimilarity* is a bisimulation iff there is a largest bisimulation (in the sense of $\subseteq$). 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

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

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 $S \times S' = \underline{\leftrightarrow}_0 \supseteq \underline{\leftrightarrow}_1 \supseteq \cdots \supseteq \underline{\leftrightarrow}_\omega \supseteq \underline{\leftrightarrow}_{\omega+1} \supseteq \cdots \supseteq \underline{\leftrightarrow}$, where $\underline{\leftrightarrow}_\omega = \bigcap_{n<\omega} \underline{\leftrightarrow}_n$ is a conjunction of "finitary" predicates $\underline{\leftrightarrow}_n$. That is to say, $\underline{\leftrightarrow}_\omega$ can be captured by a finitary formal language, as spelled out in Hennessy and Milner [26]. The equivalence $\underline{\leftrightarrow}_\omega$ need not, however, coincide with $\underline{\leftrightarrow}$, as demonstrated below (where $|L| = 1$).



This notorious pair is surely too trivial to ignore! It is easy enough to repair the Hennessy-Milner characterization above (so as to apply to $\underline{\leftrightarrow}$ in general) by allowing infinitary disjunctions (and conjunctions), but then the question arises as to what is taken for granted by building these infinitary constructs into the logic. (The measures studied in section 4 yield, among other information, bounds on the infinitary constructs required.)

Transition systems typically have obvious "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 *pointed* transition systems $(\langle L, S, \rightarrow\rangle, s_0)$ and $(\langle L, S', \rightarrow'\rangle, s'_0)$ are *bisimilar*, again writing $(\langle L, S, \rightarrow\rangle, s_0) \underline{\leftrightarrow} (\langle L, S', \rightarrow'\rangle, s'_0)$.

## 2 Some basic results and related work

A transition system $\langle L, S \rightarrow\rangle$ can be viewed as the first-order model $\langle L \cup S, L, \rightarrow\rangle$ over the signature $\{\dot{L}, \rightsquigarrow\}$ consisting of a unary relation symbol $\dot{L}$ (for the labels) and a ternary relation symbol $\rightsquigarrow$ (for $\rightarrow$). Passing to the case where $L$ is a singleton $\{l\}$, it is standard practice in modal logic (e.g., van Benthem [10]) to study the transition system $\langle\{l\}, S, \rightarrow\rangle$ as the first-order model $\langle S, \{(s, s') \mid s \xrightarrow{l} s'\}\rangle$ over the signature $\{R\}$ consisting of a binary relation symbol $R$. Such $\{R\}$-models are called *Kripke frames*, and can be expanded into *Kripke models* for a propositional modal language over propositional letters $p, q, \ldots$ by introducing interpretations for unary predicate symbols $U_p, U_q, \ldots$ (marking the states on which the corresponding letters $p, q, \ldots$ are interpreted to be true). The framework of *Kripke semantics* then specifies a translation mapping a formula of the propositional modal language into a first-order $\{R, U_p, U_q, \ldots\}$-formula over some fixed free variable $x$ (standing for a state) — e.g.,

$$\Box(p \;\&\; \Diamond\Box q) \;\;\mapsto\;\; (\forall y R^{-1} x) \;\; U_p(y) \;\&\; (\exists z R^{-1} y)(\forall u R^{-1} z) \; U_q(u) \;.$$

Now, for a suitable notion of *bisimulation invariance*, it turns out that

**Theorem A** (van Benthem [10]) *A first-order $\{R, U_p, U_q, \ldots\}$-formula with one free variable is invariant for bisimulation iff it is logically equivalent to the translation of some modal formula.*

Looking more closely at the translation of modal formulas, observe that variables can be "re-used" so that, for example, the translation of $\Box(p \;\&\; \Diamond\Box q)$ can be rewritten as $(\forall y R^{-1} x) \;\; U_p(y) \;\&\; (\exists x R^{-1} y)(\forall y R^{-1} x) \; U_q(y)$, requiring only two variables $x, y$, free or bound. Generalizing the notion of a bisimulation to that of an *n-simulation*, one can show

**Theorem B** (e.g., van Benthem [11]) *A first-order $\{R, U_p, U_q, \ldots\}$-formula with free variables $x_1, \ldots, x_n$ is invariant for n-simulation iff it can be written using only the $n$ variables $x_1, \ldots, x_n$ free or bound.*

Whatever satisfaction Theorems A and B may give, the reader is entitled to ask what does propositional modal logic have to do with our present concerns? Quite a bit, according to van Benthem, van Eijck and Stebletsova [9]. The Hennessy-Milner [26] characterization of $\underline{\leftrightarrow}_\omega$ is based on what is essentially a propositional modal logic, the only difference (beyond notation) being that many labels are required (and, in fact, one propositional letter will suffice). The theorems above adapt easily to this case, suggesting (together) that the propositional modal language corresponds, via the notion of a bisimulation, to a restricted 2-variable fragment of the first-order language over $\{\dot{L}, \leadsto, U_p, U_q, \ldots\}$.

Setting aside, however, the propositional modal language, and examining the first-order language $\{\dot{L}, \leadsto\}$ of transition systems directly, it is easy to see that bisimilarity, in fact, exceeds first-order logic. As pointed out in van Benthem and Bergstra [8] (using an argument that appeared already in Fernando [21], and to which we will return), bisimilarity cannot be defined by an infinite set of first-order sentences. This is strengthened by the following theorem, which considers bisimulations on a transition system (rather than on a pair of different ones), viewed as a first-order $\{\dot{L}, \leadsto\}$-model.

**Theorem A′.** *Bisimilarity is not preserved under elementary substructures. That is, there is a transition system with non-bisimilar states $a$ and $b$ such that an elementary extension of that transition system can be constructed over which $a$ and $b$ are bisimilar.*

**Proof.** We use the following easy facts.

> **Fact 0.** *$\underline{\leftrightarrow}_\omega$ is the conjunction of an infinite set of first-order $\{\dot{L}, \leadsto\}$-definable predicates $\sim_n$ (for every $n < \omega$) expressing $\underline{\leftrightarrow}_n$.*
>
> **Fact 1.** *Fix a transition system $\langle L, S, \rightarrow\rangle$, and let $s, s' \in S$. Then $s \underline{\leftrightarrow}_\omega s'$ iff whenever $s' \xrightarrow{l} t$, the set of $(\{\dot{L}, \leadsto\} \cup \{\dot{s}, \dot{t}, \dot{l}\})$-formulas*
>
> $$\Phi_{s,t,l}(x) \quad = \quad \{\dot{s} \overset{\dot{l}}{\leadsto} x\} \ \cup \ \{\dot{t} \sim_n x \mid n < \omega\}$$
>
> *is finitely satisfiable in the $(\{\dot{L}, \leadsto\} \cup \{\dot{s}, \dot{t}, \dot{l}\})$-model $(\langle L \cup S, L, \rightarrow\rangle, s, t, l)$.*
>
> **Fact 2.** *Every transition system has an elementary extension over which $\underline{\leftrightarrow} = \underline{\leftrightarrow}_\omega$.*

By Fact 0, it follows that the transition system described by the theorem cannot validate $\underline{\leftrightarrow} = \underline{\leftrightarrow}_\omega$ (and, in particular, the transition system must support infinite branching). Accordingly, take a transition system with $\underline{\leftrightarrow}_\omega - \underline{\leftrightarrow} \neq \emptyset$. Now, choose $(a, b) \in \underline{\leftrightarrow}_\omega - \underline{\leftrightarrow}$, and appeal to Fact 2 (a consequence of Fact 1 and compactness) and Fact 0 to obtain the required elementary extension. $\dashv$

Theorem A′ is technically similar to Theorem A in that the proofs of both involve saturation (well-known to close off suitable forms of induction — or in this case, co-induction — at $\omega$; see the discussion of Gandy's theorem in the concluding section). The significance of Fact 2 (in the proof of Theorem A′) is limited by the failure of an elementary extension to respect (in general) $\underline{\leftrightarrow}_\alpha$ for $\alpha > \omega$. (The stronger notion of an "end" extension is needed to preserve bisimilarity.) Put plainly, if a process is understood to be given by its set of transitions, then it is hardly surprising that a process becomes a second-order concept *not* preserved under elementary extensions.

Turning next to Theorem B, the partial isomorphisms (and Ehrenfeucht-Fraisse games) lying behind the $n$-simulations can, under a modified setting, be employed to build an isomorphism (rather than merely to establish elementary equivalence). The modification is based on introducing individuals (packaged in a first-order model) rather than abstracting them away as in propositional modal logic. More precisely,

**Example 1.** Fix an infinite set $X$ of variables, and a signature $\sigma$. Let $A_\sigma$ be the set of "atomic programs" $x :=?$ and $\varphi?$, where $x \in X$, and $\varphi$ is an atomic $\sigma$-formula (or an equation) with free variables from $X$. Given a $\sigma$-model $\mathbf{M}$ with universe $M$, let $S_M$ be the set of functions — henceforth called *$M$-valuations* — from finite subsets of $X$ into $M$, and let $[\![\mathbf{M}]\!]$ be the pointed transition system $(\langle A_\sigma, S_M, [\![\cdot]\!]\rangle, \emptyset)$, with initial state the empty function $\emptyset$, and where for $d, d' \in S_M$,

$$\begin{array}{rll} d[\![x :=?]\!]d' & \text{iff} & x \in \text{dom}(d) \text{ and } d = d' \text{ except possibly on } x \\ d[\![\varphi?]\!]d' & \text{iff} & d = d' \text{ and } \mathbf{M} \models \varphi[d] \end{array}$$

(the intuition being that the random assignment $x :=?$ reads input, and the test $\varphi?$ checks that $\varphi$ holds at the present state). As observed in Fernando [19], a bisimulation between $[\![\mathbf{M}]\!]$ and $[\![\mathbf{N}]\!]$ is simply a partial isomorphism set (defined, for example, in p. 97 of Keisler [28]) between $\mathbf{M}$ and $\mathbf{N}$. Thus, for countable $\mathbf{M}$ and $\mathbf{N}$, $[\![\mathbf{M}]\!] \leftrightarrow [\![\mathbf{N}]\!]$ iff $\mathbf{M} \cong \mathbf{N}$ (iff $[\![\mathbf{M}]\!] \cong [\![\mathbf{N}]\!]$). Or, in case $\mathbf{M} = \mathbf{N}$ is countable, $\omega$-homogeneity (again, see, for example, Keisler [28]) turns a bisimulation into an automorphism. More generally, an elementary "back-and-forth" construction from model theory yields

**Theorem B$'$.** *For all $\sigma$-models $\mathbf{M}$ and $\mathbf{N}$, if an $[\![\mathbf{M}]\!]$-state $d_M$ is bisimilar to an $[\![\mathbf{N}]\!]$-state $d_N$, then $dom(d_M) = dom(d_N)$, and, moreover, if $M$ and $N$ are countable, then the correspondence*

$$d_M(x) \quad \mapsto \quad d_N(x)$$

*(for all $x \in dom(d_M)$) extends to an isomorphism between $\mathbf{M}$ and $\mathbf{N}$.*

Hence, in the terminology of van Benthem [12], the rather meager "programming repertoire" $A_\sigma$ guarantees *safety for bisimulations* — i.e., any expansion of the label set $A_\sigma$ in which the transitions are defined "uniformly" from a countable first-order model will preserve bisimulations. [Proof: given a transition system on valuations, take its restriction to the above label set, and pass the isomorphism between $\mathbf{M}$ and $\mathbf{N}$, described by Theorem B$'$, on to the original transition system.] For an illustration of what is meant by "uniform" (beyond the requirement that isomorphic objects in $\sigma$-models map to isomorphic states of the transition systems defined from the $\sigma$-models), see the next section, where an additional feature of effectiveness is introduced. That section considers more carefully transitions that are relevant to the study of operational semantics in the following sense.

Going back (at least) to Turing, mechanical computation has been characterized by transitions $c \rightarrow c'$ between "configurations" $c$ and $c'$ subject to a certain set of rules. The transitions will be labelled soon enough, but for the moment, let us take the transitions to be unlabelled. Let us decompose a configuration $c$ into a "data" component $d$ and a control or "program" component $p$, whence the transition $c \rightarrow c'$ becomes $(d,p) \rightarrow (d',p')$.

**Example 2.** The transition system $[\![\mathbf{M}]\!]$ (for a fixed $\sigma$-model $\mathbf{M}$) of Example 1 gives a set of transitions $(d,a) \rightarrow (d',\surd)$ for $d,d' \in S_M$, $a \in A_\sigma$ and $d[\![a]\!]d'$. (The fresh symbol $\surd$ denotes the "terminal" or "null" control state.) This transition set can be extended by closing the label set $A_\sigma$ under various program constructs. Examples include sequential composition ;, non-deterministic choice $+$ and Kleene star $\cdot^*$, which are analyzed in dynamic logic (e.g., Harel [25]) compositionally over programs conceived as input/output relations. That is to say, such programs are determined completely by transitions into $(d',\surd)$. In general, however, "intermediate" computational states in a possibly non-terminating computation may be of interest as well; in other words, we might also consider transitions $(d,p) \rightarrow (d',p')$ where $p' \neq \surd$ is a "job" that remains to be done. Such transitions are convenient (if not essential) for a construct such as interleaving $\|$, which might be introduced subject to the rules

$$\frac{(d,p) \rightarrow (d',p')}{(d,p\|p'') \rightarrow (d',p'\|p'')} \qquad \frac{(d,p\|p'') \rightarrow (d',p')}{(d,p''\|p) \rightarrow (d',p')} \qquad \frac{(d,p) \rightarrow (d',p'\|p'')}{(d,p) \rightarrow (d',p''\|p')} .$$

Other rules might include

$$\frac{}{(d,\varphi?) \rightarrow (d,\surd)}\ \mathbf{M} \models \varphi[d] \qquad \frac{}{(d,x :=?) \rightarrow (d',\surd)}\ x \in \mathrm{dom}(d') \text{ and } d = d' \text{ except possibly on } x$$

$$\frac{(d,p) \rightarrow (d',\surd)}{(d,p;p') \rightarrow (d',p')} \qquad \frac{(d,p) \rightarrow (d',\surd)}{(d,p+p') \rightarrow (d',\surd)} \qquad \frac{(d,\mathit{skip}+p;p^*) \rightarrow (d',p')}{(d,p^*) \rightarrow (d',p')} .$$

The rules above are not "complete", but meant simply to provide some intuition. (Note that, from the point of view of predicate logic, "rule" here is better read as "axiom" — and the presentation of such axioms is unfortunate — in that the premiss should be interpreted locally over a fixed model, rather than globally over a family of models.) The treatment of $\cdot^*$ generalizes easily to solutions to recursive equations. Synchronization on actions can also be accomodated, by adding the actions (on which processes synchronize) to the underlying first-order model. See Fernando [22] for more details. [end of example]

Now, to analyze programs relative to data, it is useful to rewrite the transition $(d,p) \rightarrow (d',p')$ as

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

with the idea of using bisimilarity on programs $p, p'$ as a notion of program equivalence. This is taken up in section 4, which investigates further the co-inductive construction of bisimilarity, and is in this sense a natural sequel to Theorem A$'$ above. Before, however, carrying out an analysis that takes data for granted (relegating it to labels of a transition system, where, in fact, it is commonly abstracted away), let us consider more carefully just what is taken for granted, and see if we can come up with a story different from that offered by Theorem B$'$.

## 3 Bisimulations and data dependence

To analyze the dependence of transitions on data, let us rewrite the transition $(d,p) \rightarrow (d',p')$ as

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

employing the notion of bisimulation on such transitions. Observe that the transition systems $[\![\mathbf{M}]\!]$ and their extensions defined in the previous section are of this form. The constructions considered in van Benthem [12] apply most naturally to this case where, as already mentioned, safety for bisimulations becomes automatic in view of Theorem B$'$. Automatic, that is, so long as the constructions are "uniform", as is the case for the following class of constructions that we now turn to.

Fix a set $L$ of labels, a signature $\sigma$, and a countable set $X$ of variables. Consider a function mapping a $\sigma$-model $\mathbf{M}$ (with universe $M$) into a transition relation $\rightarrow_{\mathbf{M}} \subseteq S_M \times L \times S_M$ (where $S_M$ is the set of $M$-valuations — i.e., functions from a finite subset of $X$ into $M$). Given a $\sigma$-model $\mathbf{M}$, a label $l \in L$, and $M$-valuations $d : X_0 \rightarrow M$ and $d' : Y_0 \rightarrow M$, call a first-order $\sigma$-formula $\varphi$ with free variables from $X_0 + Y_0$ a *record of* $d \overset{l}{\rightarrow}_{\mathbf{M}} d'$ if

(i) $\mathbf{M} \models \varphi[d + d']$, and

(ii) for all $\sigma$-models $\mathbf{N}$ and $N$-valuations $d_N : X_0 \rightarrow N$ and $d'_N : Y_0 \rightarrow N$, if $\mathbf{N} \models \varphi[d_N + d'_N]$ then $d_N \overset{l}{\rightarrow}_{\mathbf{N}} d'_N$.

(The intuition is that $X_0$ records the input, and $Y_0$ the "output", except that the "output" may actually refer to an intermediate computational state.)

**Illustration.** For $L$ given by regular programs in quantified dynamic logic, records of a particularly simple (linear) form can be obtained, due to a reduction reminiscent of the Kleene normal form theorem. (A more general result guaranteeing the existence of records will be proved shortly.) Given a regular program $p$ and a transition $\emptyset \overset{p,\surd}{\rightarrow}_{\mathbf{M}} d$, written $\emptyset[\![p]\!]_{\mathbf{M}}d$ for notational convenience, we can extract a finite sequence $p_1;p_2;\ldots;p_n$ of tests and assignments (from the definition of $[\![p]\!]$) such that

(i) $\emptyset[\![p_1;p_2;\ldots;p_n]\!]_{\mathbf{M}}s$, and

(ii) for every $\sigma$-model $\mathbf{N}$ and $N$-valuation $s'$, if $\emptyset[\![p_1;p_2;\ldots;p_n]\!]_{\mathbf{N}}s'$, then $\emptyset[\![p]\!]_{\mathbf{N}}s'$.

Then, for $i = 1,\ldots n$, let $I_i$ be the set $\{x_1,\ldots,x_{k_i}\}$ of variables in $X_0$ mentioned in $p_1;\ldots;p_i$, and let $x^i_1,\ldots,x^i_{k_i}$ be fresh variables, the intention being that $x^i_j$ represents the value of $x_j$ after $p_i$ is executed. The transition $\emptyset[\![p]\!]_{\mathbf{M}}s$ is then recorded by

$$\exists x^1_1 \cdots \exists x^1_{k_1} \cdots \exists x^n_1 \cdots \exists x^n_{k_n} \quad \bigwedge_{1 \leq j \leq k_n} x_j = x^n_j \;\; \& \bigwedge_{1 < i \leq n} \varphi_i ,$$

where for $1 < i \leq n$, $\varphi_i$ relates $x^{i-1}_1,\ldots,x^{i-1}_{k_{i-1}}$ to $x^i_1,\ldots,x^i_{k_i}$ after the execution of $p_i$. [end of illustration]

Next, we isolate a certain form of transition rules yielding transitions that can be recorded.

**$\Phi$-uniform rules.** Given a set $\Phi$ of $\sigma$-formulas with free variables from $X$, a rule $r$ is *$\Phi$-uniform* if it has the form

$$\frac{s_1 \xrightarrow{l_1} t_1 \quad \cdots \quad s_n \xrightarrow{l_n} t_n}{s \xrightarrow{l} t} \quad C_r(s_1, t_1, \ldots, s_n, t_n, s, t)$$

where (i) $l_1, \ldots, l_n, l \in L$, (ii) $s_1, t_1, \ldots, s_n, t_n, s, t$ are state-variables (*not* to be confused with variables in $X$, but meant rather to range over $M$-valuations, for $\sigma$-models $\mathbf{M}$), and (iii) the condition $C_r(s_1, t_1, \ldots, s_n, t_n, s, t)$ enjoys the following "uniformity" property relative to $\Phi$

> for all finite subsets $X_1, Y_1, \ldots, X_n, Y_n, X_0, Y_0$ of $X$, there is a $\sigma$-formula $\varphi \in \Phi$ with free variables among[1] $X_1 + Y_1 + \cdots + X_n + Y_n + X_0 + Y_0$ such that for every $\sigma$-model $\mathbf{M}$ and for all $d_1 : X_1 \to M, d_1' : Y_1 \to M, \ldots, d_n : X_n \to M, d_n' : Y_n \to M, d : X_0 \to M, d' : Y_0 \to M$,
>
> $$C_r(d_1, d_1', \ldots, d_n, d_n', d, d') \text{ holds} \quad \text{iff} \quad \mathbf{M} \models \varphi[d_1 + d_1' + \cdots + d_n + d_n' + d + d'] \,.$$

Observe that a $\Phi$-uniform rule $r$ is "positive" in that its premiss consists of positive clauses $s_i \xrightarrow{l_i} t_i$, plus a side-condition $C_r$ reducible to formulas from $\Phi$. A negative condition $\neg(s_i \xrightarrow{l_i} t_i)$ can be approximated by introducing a new label $\sim l_i$ and replacing $\neg(s_i \xrightarrow{l_i} t_i)$ by $s_i \xrightarrow{\sim l_i} t_i$ (borrowing the idea of "strong negation"). But one cannot expect, for instance, the negation construct

$$s \xrightarrow{\neg p} t \quad \text{iff} \quad s = t \text{ and there is } no \ s' \text{ s.t. } s \xrightarrow{p} s'$$

in van Benthem [12] to be given by $\Phi$-uniform rules, if satisfiability of formulas in $\Phi$ is r.e. (since $\neg p$ is, in the notation of dynamic logic, the test $[p]\bot?$ for the non-r.e. complement of the halting problem). On the other hand, the reader may verify that if the transitions $(d, p) \to (d', p')$ described in Example 2 are rewritten as $d \xrightarrow{p,p'} d'$, then they can be presented as theorems of $\Phi$-uniform rules, where $\Phi$ is the set of conjunctions of atomic $\sigma$-formulas (including equalities) over $X$, and the condition $C_r$ is (essentially) vacuous except on assignments, tests and *skip*. The advantage of presenting transitions in terms of $\Phi$-uniform rules is the following. Given a set $\Phi$ of $\sigma$-formulas, let $\Phi_\exists$ consist of all formulas in $\Phi$ and those obtained from it by closing under conjunction and existential quantification (as well as renaming of variables in $X$).

**Lemma 1** *(Record Property). Every transition proved from $\Phi$-uniform rules has a record in $\Phi_\exists$.*

**Proof.** Every theorem has a finite derivation tree, from which a record in $\Phi_\exists$ can be extracted: local descriptions of the tree (by formulas in $\Phi$) are glued together by conjunction, before existentially quantifying out old values of program variables. $\dashv$

The converse of Lemma 1 is even easier: transitions with records in $\Phi$ can be presented as theorems of $\Phi$-uniform rules (where the side conditions $C_r$ do all the work, and are given by the records). It appears, however, that, in practice, a transition relation is more naturally presented in terms of $\Phi$-uniform rules. Accordingly, we will work with transition systems $\mathbf{M}_{\mathcal{R}} = \langle L, S_M, \to_{\mathbf{M}} \rangle$ where $\to_{\mathbf{M}}$ consists of the theorems of a set $\mathcal{R}$ of $\Phi$-uniform rules. It will prove convenient to equate $\mathbf{M}_{\mathcal{R}}$ with the pointed transition system $(\langle L, S_M, \to_{\mathbf{M}} \rangle, \emptyset)$ where the initial state $\emptyset$ is the totally undefined function. Also, given $\sigma$-models $\mathbf{M}$ and $\mathbf{N}$ and a set $\Psi$ of $\sigma$-formulas, let us write $\mathbf{M} \equiv_\Psi \mathbf{N}$ when $\mathbf{M}$ and $\mathbf{N}$ satisfy the same $\sigma$-sentences in $\Psi$. (Note the switch from formulas to sentences — i.e., formulas with no free variables.) Since transitions are determined by records, it is natural to seek out a translation $\mathcal{L}$ on pointed transition systems so that

($\dagger$) *If $\mathcal{R}$ is a set of $\Phi$-uniform rules, then* $\mathcal{L}(\mathbf{M}_{\mathcal{R}}) \cong \mathcal{L}(\mathbf{N}_{\mathcal{R}})$ *iff* $\mathbf{M} \equiv_{\Phi_\exists} \mathbf{N}$ *iff* $\mathcal{L}(\mathbf{M}_{\mathcal{R}}) \underline{\leftrightarrow} \mathcal{L}(\mathbf{N}_{\mathcal{R}})$.

In fact, as we now show, $\mathcal{L}$ can be defined in a familiar way — essentially, the subset construction in automata theory, mapping non-deterministic finite automata to deterministic finite automata.

Given a pointed transition system $\mathbf{S} = (\langle L, S, \to \rangle, s_0)$, define for every $l \in L$ the function $[l]_\to$ : Power($S$) $\to$ Power($S$) by

$$[l]_\to(a) \quad = \quad \{t \in S \mid \exists s \in a \ s \xrightarrow{l} t\} \,.$$

[1]For subsets $X'$ and $Y'$ of $X$, the notation $X' + Y'$ is used for the disjoint sum of the sets; and for functions $d : X' \to M$ and $d' : Y' \to M$, the notation $d + d'$ is used for the function with domain $X' + Y'$ given in the obvious way by $d$ and $d'$.

Then let $\mathcal{L}(\mathrm{S})$ be the pointed transition system $((L, \mathcal{L}(S), \Rightarrow), \{s_0\})$ where $\mathcal{L}(S)$ is given inductively by

$$\frac{}{\{s_0\} \in \mathcal{L}(S)} \qquad \frac{l \in L \quad a \in \mathcal{L}(S) \quad [l]_{\rightarrow}(a) \neq \emptyset}{[l]_{\rightarrow}(a) \in \mathcal{L}(S)} ,$$

and for every $l \in L$,

$$\overset{l}{\Rightarrow} \; = \; [l]_{\rightarrow} \cap (\mathcal{L}(S) \times \mathcal{L}(S)) .$$

A simple illustration based on dynamic logic should be sufficient to clarify the definition.

**Illustration.** Let $p$ be the non-deterministic program $x := 0 + x := 1$. Over the standard model of arithmetic, $p$ sends the empty valuation $\emptyset$ either to the valuation $\{(x, 0)\}$ mapping $x$ to 0, or to the valuation $\{(x, 1)\}$ mapping $x$ to 1; i.e., in the notation of the previous illustration

$$\emptyset[\![p]\!]s \quad \text{iff} \quad s = \{(x,0)\} \text{ or } s = \{(x,1)\} .$$

Applying the operator $\mathcal{L}$ to this transition system with initial state $\emptyset$ then gives

$$\{\emptyset\} \overset{p;\surd}{\Rightarrow} a \quad \text{iff} \quad a = \{\{(x,0)\}, \{(x,1)\}\} .$$

[end of illustration]

Note that $\overset{l}{\Rightarrow}$ is a partial function on $\mathcal{L}(S)$ that is not always defined since the empty set is excluded from $\mathcal{L}(S)$ (lest the notion of a bisimulation become trivial over the image of $\mathcal{L}$). The empty set would represent an "absurd" state, indicating the failure to make a transition (in S). Such failures completely determine bisimilarity over the deterministic transition systems $\mathcal{L}(\mathrm{S})$; isomorphism (over the image of $\mathcal{L}$) captures (prima facie) a bit more structure — namely, for any sequence of labels $l_1, \ldots, l_n$, the set (viewed externally) of states in S accessible from the initial state of S by transitions labelled by $l_1, \ldots, l_n$. The logical significance of $\mathcal{L}$ is brought out partly in Fernando [19], and partly below.

Concentrating on the case of $\mathcal{L}(\mathrm{M}_{\mathcal{R}})$, let us write $[l]_{\mathrm{M}}$ for the interpretation of $l \in L$ by $\mathcal{L}(\mathrm{M}_{\mathcal{R}})$ (omitting the subscript $\mathcal{R}$ for notational simplicity). The key to establishing (†), the non-trivial part of which is isomorphism, is

**Lemma 2.** *Let $\mathcal{R}$ be a set of $\Phi$-uniform rules, $\mathrm{M} \equiv_{\Phi_\exists} \mathrm{N}$, $\{\emptyset\}[l]_{\mathrm{M}}a$, and $\{\emptyset\}[l']_{\mathrm{M}}a$. Then for every $N$-valuation $s$, if $\emptyset \xrightarrow{l}_{\mathrm{N}} s$ then $\emptyset \xrightarrow{l'}_{\mathrm{N}} s$.*

**Proof.** Assume $\emptyset \xrightarrow{l}_{\mathrm{N}} s$. Let $X_0$ be the domain of $s$, and

$$\begin{aligned} \Psi &= \{\varphi \in \Phi_\exists \mid \varphi \text{ is a record of } \emptyset \xrightarrow{l'}_{\mathrm{M}'} s' \text{ for some } \sigma\text{-model } \mathrm{M}' \text{ and } s' : X_0 \rightarrow M'\} \\ \Theta &= \{\theta \mid \mathrm{N} \models \theta[s] \text{ and } \theta \text{ or } \neg\theta \in \Phi_\exists \text{ with free variables from } X_0\} . \end{aligned}$$

By the definition of a record, it suffices to produce a $\varphi \in \Psi \cap \Theta$. So, by the definition of $\Theta$, we need only demonstrate the consistency of $\{\bigvee \Psi\} \cup \Theta$. But now let $\hat{\varphi} \in \Phi_\exists$ be a record of $\emptyset \xrightarrow{l}_{\mathrm{N}} s$, and fix a finite $\Delta \subset \Theta$. Then $\hat{\varphi} \& \bigwedge \Delta$ is satisfied by $\mathrm{M}$ under some $t : X_0 \rightarrow M$, since $\mathrm{M} \equiv_{\Phi_\exists} \mathrm{N}$. Moreover, since $\{\emptyset\}[l]_{\mathrm{M}}a$ and $\{\emptyset\}[l']_{\mathrm{M}}a$ (by assumption), it follows (appealing again to the definition of a record) that there is some $\varphi \in \Psi$ satisfied by $\mathrm{M}$ under $t$. In other words, for every finite piece $\Delta \subset \Theta$, there is a $\varphi \in \Psi$ such that $\varphi \& \bigwedge \Delta$ is satisfiable. Thus, the General Omitting Types Theorem given in p. 108 of Keisler [27] yields the consistency of $\{\bigvee \Psi\} \cup \Theta$, as required. $\dashv$

**Lemma 3.** *Let $\mathcal{R}$ be a set of $\Phi$-uniform rules, $\mathrm{M} \equiv_{\Phi_\exists} \mathrm{N}$, $\{\emptyset\}[l]_{\mathrm{M}}a$, and $\{\emptyset\}[l]_{\mathrm{N}}a'$. Then for all $l' \in L$,*

$$\{\emptyset\}[l']_{\mathrm{M}}a \quad \textit{iff} \quad \{\emptyset\}[l']_{\mathrm{N}}a' .$$

**Proof.** It suffices to prove one direction, say $\Rightarrow$, of the equivalence, appealing to symmetry for the other. But then $\Rightarrow$ is immediate from two applications of Lemma 2. $\dashv$

**Theorem 4.** *Let $\mathcal{R}$ be a set of $\Phi$-uniform rules. If $\mathbf{M} \equiv_{\Phi_\exists} \mathbf{N}$, then $\mathcal{L}(\mathbf{M}_\mathcal{R}) \cong \mathcal{L}(\mathbf{N}_\mathcal{R})$.*

**Proof.** Let $R$ be the relation consisting of all pairs $(a, a')$ of $\mathcal{L}(\mathbf{M}_\mathcal{R})$-states $a$ and $\mathcal{L}(\mathbf{N}_\mathcal{R})$-states $a'$ for which there is a finite sequence $l_1, \ldots, l_n$ of labels in $L$ such that $\{\emptyset\}[l_1]_\mathbf{M} \circ \cdots \circ [l_n]_\mathbf{M} a$ and $\{\emptyset\}[l_1]_\mathbf{N} \circ \cdots \circ [l_n]_\mathbf{N} a'$ (where $\circ$ is relational composition). To see that $R : \mathcal{L}(\mathbf{M}_\mathcal{R}) \cong \mathcal{L}(\mathbf{N}_\mathcal{R})$, assuming $\mathbf{M} \equiv_{\Phi_\exists} \mathbf{N}$, observe that from the preceding lemma, $R$ is a function from $\mathcal{L}(\mathbf{M}_\mathcal{R})$-states to $\mathcal{L}(\mathbf{N}_\mathcal{R})$-states, and, writing $f$ for that function,

$$a_0[l]_\mathbf{M} a_1 \quad \text{iff} \quad f(a_0)[l]_\mathbf{N} f(a_1) \; ,$$

assuming without loss of generality that $L$ is closed under sequential composition ;, and that $\emptyset \xrightarrow{l;l'} s$ iff $(\exists t)\ \emptyset \xrightarrow{l} t$ and $t \xrightarrow{l'} s$. ⊣

**Corollary 5.** *Assume $\Phi$ is closed under renaming of variables (in $X$), and that $\mathcal{R}$ is a set of $\Phi$-uniform rules, including rules for sequential composition, random assignments $x :=?$ for $x \in X$, and tests $\varphi?$ for all $\varphi \in \Phi$. For $\sigma$-models $\mathbf{M}$ and $\mathbf{N}$, the following are equivalent.*

(i) $\mathbf{M} \equiv_{\Phi_\exists} \mathbf{N}$.

(ii) $\mathcal{L}(\mathbf{M}_\mathcal{R}) \cong \mathcal{L}(\mathbf{N}_\mathcal{R})$.

(iii) $\mathcal{L}(\mathbf{M}_\mathcal{R}) \underline{\leftrightarrow} \mathcal{L}(\mathbf{N}_\mathcal{R})$.

**Proof.** '(i) implies (ii)' is Theorem 4; '(ii) implies (iii)' is trivial ($\cong$ always implies $\underline{\leftrightarrow}$); and '(iii) implies (i)' is a consequence of the fact that all of $\Phi_\exists$ can be programmed using tests, sequential composition and random assignments (the latter two constructs closing $\Phi$ under conjunction and existential quantification). ⊣

Theorem 4 runs against the suggestion from Theorem B′ that mechanical transitions generally depend on *more* than the first-order theory of the data model, although perhaps the underlying logic is best associated with *positive existential induction* (Aczel [3], section 3.2) in view of the Record Property, Lemma 1.

At any rate, by equating bisimilarity with isomorphism under $\mathcal{L}$, Corollary 5 reduces $\mathcal{L}(\mathbf{S})$ to the sequences $l_1, \ldots, l_n$ of labels for which the initial state of $\mathbf{S}$ fails to make a transition. The most technically involved part of the argument above is the omission of types in Lemma 2. The idea of working with "small" models contrasts with the appeal in Theorems A and A′ to "large" (saturated) models (borrowing the "terminology" from Keisler [28]). Omitting types arguments often arise when showing completeness for an $\omega$-rule. Making a turn towards the opposite direction, the inadequacy of an $\omega$-rule is exposed in the next section, where the point is that certain second-order types cannot be omitted. Whereas the dependence of transitions on data is analyzed above by internalizing the non-determinism of transitions in states taken as sets (according to the logical translation $\mathcal{L}$), the notion of a bisimulation is applied in the next section to give an equivalence on programs, leading to a different use of sets (i.e., as equivalence classes).

## 4 Bisimulations and generalized extensionality

A fundamental problem in programming language semantics is the notion of identity on programs. On the one hand, equating a program with its text is not terribly enlightening. On the other hand, reducing a program to the input/ouput relation it computes abstracts away how that relation is computed — which is often of some interest. A notion of equivalence between programs can be defined relative to a fixed level of abstraction specified by transitions between mechanical configurations by reformulating the transition $(d, p) \rightarrow (d', p')$ as

$$p \xrightarrow{d,d'} p' \; ,$$

and then forming bisimilarity over this transition system. In process semantics as well as in so-called structural operational semantics (Plotkin [32]), data is typically abstracted away, and the label $(d, d')$ replaced by an "*atomic* action" $a$ (presumably taking $d$ to $d'$; see, e.g., Fernando [22]). Before abstracting away the data (and the labels), however, let us observe that bisimilarity subsumes input/output equivalence. In case all transitions represent completed input/output computations (i.e., in case all transitions end at $\surd$), then bisimilarity is simply input/output equivalence. Otherwise, transition predicates $\rightarrow_*$ and $\rightarrow_{at}$ might be defined inductively according to

$$\frac{p \overset{d,d'}{\rightarrow} q}{p \overset{d,d'}{\rightarrow}_* q} \qquad \frac{p \overset{d,d'}{\rightarrow}_* q \quad q \overset{d',d''}{\rightarrow} r}{p \overset{d,d''}{\rightarrow}_* r} \qquad \frac{p \overset{d,d'}{\rightarrow}_* \surd}{p \overset{d,d'}{\rightarrow}_{at} \surd},$$

whence input/output equivalence amounts to bisimilarity $\underline{\leftrightarrow}_{at}$ relative to $\rightarrow_{at}$. It is natural to view $\cdot_*$ and $\cdot_{at}$ as maps on transition systems, providing a translation between different levels of abstraction (of which input/output transitions constitute a coarse example, taken as a starting point for an "effective" category-theoretic study in Fernando [20]). A similar translation reduces so-called trace equivalence to bisimilarity. More generally, given that a number of process equivalences amount to an equality between certain sets associated with the processes, it is noteworthy that bisimilarity can be viewed as an equality predicate on sets, as developed at length in Aczel [4]. These sets can, of course, be somewhat abstract — as with equivalences based on logical characterizations in some language (where the sets in question are formed out of the formulas satisfied by the processes). But the reduction to bisimilarity need not always require a heroic leap of imagination, as illustrated above. At any rate, one might certainly ask whether some sort of translation exists in principle, before worrying about just exactly how it looks. Accordingly, the slogan

*the scope of the notion of a bisimulation rests on its complexity*

can be construed in the rigorous sense in which logical complexity is measured by the existence of translations. Turning then to these precise measures, note that as input/output equivalence is, in general, $\Pi^0_2$-complete, its reduction to bisimilarity requires a context that can support that complexity. By contrast, bisimilarity is commonly taken to be at worst $\Pi^0_1$ in process semantics (e.g., Bloom, Istrail and Meyer [13]) by working with a transition system in which only finitely many transitions are possible from a state (whence $\underline{\leftrightarrow}_\omega = \underline{\leftrightarrow}_{\omega+1}$), and that finite set is computable (whence $\underline{\leftrightarrow}_\omega$ is $\Pi^0_1$). The reduction to input/output equivalence above need not involve infinite branching (on any fixed label), but the transitive closure in $\cdot_*$ may generate undecidable transitions (that will, however, still be r.e. provided $\rightarrow$ is).

In fact, as we will soon see, the transition predicate can be kept to a very low mechanical complexity (i.e., linear time!) while still blowing up bisimilarity well beyond $\Pi^0_2$. Bisimilarity is manifestly $\Sigma^1_1$ relative to its transition predicate, but whether this bound is optimal is stated to be open in Darondeau [16] (p. 229). The rest of this section is devoted to establishing[2]

(‡) There is a transition predicate of low complexity over which bisimilarity is *not* $\Pi^1_1$.

As far as (‡) is concerned, it turns out that we can make do with a singleton label set — which is to say that labels are a notational nuisance. Accordingly, *in the remainder of the present section, transition predicates will be understood to be binary relations on* $\omega$, *and bisimilarity relative to such a relation* $\leadsto$ *will be construed as that defined over the transition system* $\langle\{*\}, \omega, \{(n, *, m) \mid n \leadsto m\}\rangle$. A fact complicating (‡) is that bisimilarity is also $\Pi^1_1$ (whence $\Delta^1_1$) over well-founded transition predicates — a result implicit in Barwise, Gandy and Moschovakis [6] (p. 115), as well as in van Benthem and Bergstra [8] (p. 27). Hence, we will also consider non-well-founded transition predicates. Moreover, turning to the co-inductive characterization of bisimilarity, note that for bisimilarity to fall outside $\Pi^1_1$, the co-inductive construction must not terminate at any stage named by a recursive ordinal. So, keeping all these ordinals in view, consider the Church-Kleene system $\mathcal{O}$ of ordinal notations, which, for the purposes of (‡), can be presented as follows.

Fix a standard enumeration $\{\langle e\rangle\}_{e\in\omega}$ of unary primitive recursive functions, and define the transition predicate $\rightarrow$ inductively by the following rules, where $n, e, m$ and $k$ range over $\omega$

$$\frac{}{2^n \rightarrow n} \qquad\qquad \frac{n \rightarrow m \quad m \rightarrow k}{n \rightarrow k}\ \text{(trans)}$$

[2]In doing so, it refines, corrects and extends part of the technical report Fernando [21].

$$\frac{\langle e\rangle(1) \rightarrow \langle e\rangle(0)}{3\cdot 5^e \rightarrow \langle e\rangle(0)} \qquad\qquad \frac{3\cdot 5^e \rightarrow \langle e\rangle(n) \qquad \langle e\rangle(n+2) \rightarrow \langle e\rangle(n+1)}{3\cdot 5^e \rightarrow \langle e\rangle(n+1)}$$

The set $\mathcal{O}$ of *ordinal notations* consists of all natural numbers $a$ such that there is *no* sequence $\{a_i\}_{i<\omega}$ for which

$$a \rightarrow a_0 \rightarrow a_1 \rightarrow a_2 \rightarrow \cdots .$$

The idea is that an $a \in \mathcal{O}$ names the recursive ordinal $|a|$ given by

$$|a| \;=\; \sup\,\{|b|+1 \mid a \rightarrow b\}\;.$$

While we will have no reason to assign finite ordinals unique notations, the rule scheme (trans) and the premisses of the rules for $3\cdot 5^e \rightarrow \langle e\rangle(n)$ are introduced to secure

**Lemma 6.** *The relation $\rightarrow$ is transitive, and, moreover, for every $a \in \omega$, if $\rightarrow$ restricted to $\{a\} \cup \{b \mid a \rightarrow b\}$ is irreflexive, then $\rightarrow$ restricted to $\{a\} \cup \{b \mid a \rightarrow b\}$ is, in fact, a linear order.*

This technical condition pushes through the limit clause of the induction argument for

**Lemma 7** (essentially Klop [29]). *For every $a \in \mathcal{O}$, and every $b \in \omega$,*

$$a \underline{\leftrightarrow} b \quad \textit{iff} \quad b \in \mathcal{O} \textit{ and } |b| = |a|\;.$$

As indicated in the previous result, the transition system we are dealing with is very close to the "ordinal processes" of Klop [29] — i.e., transition systems given by a singleton label set, a successor ordinal as the state set, and the ordering $>$ as the transition predicate. The crucial difference is that the states in the present transition system may be non-well-founded — a property that we exploit next. Following Feferman and Spector [17], define a superset $\mathcal{O}^*$ of $\mathcal{O}$ as the set of natural numbers $a$ such that there is *no hyperarithmetic*[3] sequence $\{a_i\}_{i<\omega}$ for which $a \rightarrow a_0 \rightarrow a_1 \rightarrow a_2 \rightarrow \cdots$. Now, the essential point is that from Feferman and Spector [17], we know that $\mathcal{O}^* \not\subseteq \mathcal{O}$ (because $\mathcal{O}^*$ is $\Sigma^1_1$, whereas $\mathcal{O}$ is not) and that

$$(*) \qquad \forall \hat{a} \in \mathcal{O}^* - \mathcal{O} \qquad \{|a| \mid a \in \mathcal{O} \text{ and } \hat{a} \rightarrow a\} = \omega_1^{CK}\;,$$

(where $\omega_1^{CK}$ is the first non-recursive ordinal) whence an induction argument (on co-induction) yields

**Lemma 8.** *For all $a, a' \in \mathcal{O}^* - \mathcal{O}$, $a \underline{\leftrightarrow} a'$.*

**Proof.** Argue by induction on $\alpha$ that for all $\alpha$ and $a, a' \in \mathcal{O}^* - \mathcal{O}$, $a \underline{\leftrightarrow}_\alpha a'$. The base case $\alpha = 0$ and the inductive step where $\alpha$ is a limit are trivial. So let $\alpha = \beta + 1$. For all $a \in \mathcal{O}^*$, and $b \in \omega$, observe that $a \rightarrow b$ implies $b \in \mathcal{O}$ or $b \in \mathcal{O}^* - \mathcal{O}$. Appealing to Lemma 7 and $(*)$ for the former case (i.e., $b \in \mathcal{O}$), and the induction hypothesis for the latter case, conclude that for all $a, a' \in \mathcal{O}^* - \mathcal{O}$, $a \underline{\leftrightarrow}_\alpha a'$. $\dashv$

Lemma 8 is a more complicated form of an argument in van Benthem and Bergstra [8] (p. 12) proving that bisimilarity cannot be defined by an infinite set of first-order formulas. The complication consists of the introduction of infinite branching (which is required to establish $\underline{\leftrightarrow} \notin \Pi^1_1$; otherwise $\underline{\leftrightarrow} = \underline{\leftrightarrow}_\omega \in \Delta^1_1$) and of the possibility of non-hyperarithmetic descending sequences (for which, thankfully, we already know $(*)$).

Next, given an $\hat{a} \in \mathcal{O}^* - \mathcal{O}$, observe that $\rightarrow$ restricted to $\{\hat{a}\} \cup \{a \mid \hat{a} \rightarrow a\}$ is a linear order (by Lemma 6). Following the well-known reduction of r.e. orders to primitive recursive orders, define a primitive recursive function $f_{\hat{a}}$ by

$$f_{\hat{a}}(n) \;=\; \begin{cases} \hat{a} & \text{if } P^n_{\hat{a}} = \emptyset \\ \text{the } a \text{ s.t. the least } p \in P^n_{\hat{a}} \text{ proves } \hat{a} \rightarrow a & \text{otherwise} \end{cases}$$

where (under a natural Gödel numbering of proofs with a primitive recursive proof predicate)

$$\begin{aligned} P^n_{\hat{a}} \;=\; \{p < n \mid\ & p \text{ proves } \hat{a} \rightarrow a \text{ for some } a \neq f_{\hat{a}}(0), \ldots, f_{\hat{a}}(n-1) \text{ s.t.} \\ & (\forall i < n)\ (\exists q < n)\ \ q \text{ proves } a \rightarrow f_{\hat{a}}(i) \text{ or } q \text{ proves } f_{\hat{a}}(i) \rightarrow a\}\;. \end{aligned}$$

[3]For orientation, a well-known result of Kreisel's is that the hyperarithmetic sets form the least $\omega$-model of $\Delta^1_1$-comprehension (recalling Kleene's classic theorem equating $\Delta^1_1$ with hyperarithmetic sets). The present section shows that the notion of a bisimulation requires more types (in contrast to the previous section where types were omitted).

It follows from Lemma 6 that the image of $f_{\hat{a}}$ is precisely $\{\hat{a}\} \cup \{a \mid \hat{a} \to a\}$. Now, let $\to_{\hat{a}}$ be the primitive recursive predicate

$$s \to_{\hat{a}} s' \quad \text{iff} \quad (\exists p < \max(s, s'))\ p \text{ proves } f_{\hat{a}}(s) \to f_{\hat{a}}(s') \ ,$$

and consider the $\Pi^1_1$-path

$$Z_{\hat{a}} \quad = \quad \{a \in \mathcal{O} \mid \hat{a} \to a\}$$

through $\mathcal{O}$ that $\hat{a}$ gives. The following program consults an oracle for bisimilarity $\underline{\leftrightarrow}_{\hat{a}}$ over $\to_{\hat{a}}$ to decide (by the construction of $f_{\hat{a}}$ and Lemma 8) whether or not $a \in Z_{\hat{a}}$

```
check if $\hat{a} \to a$;                { This is a $\Sigma^0_1$ problem easily reducible to bisimilarity. }
if not, then $a \notin Z_{\hat{a}}$;
otherwise, search the $n$ such that $f_{\hat{a}}(n) = a$, concluding that $a \in Z_{\hat{a}}$ iff $n \underline{\leftrightarrow}_{\hat{a}} 0$.
```

By Friedman [23], $\hat{a}$ can be chosen such that $\mathcal{O}$ is recursive in $Z_{\hat{a}}$. But $\mathcal{O}$ is $\Pi^1_1$-complete (under many-one reductions, no less) and bisimilarity is $\Sigma^1_1$. Thus,

**Theorem 9.** *There is a primitive recursive transition predicate over which bisimilarity is $\Sigma^1_1$-complete under Turing reductions (whence non-$\Pi^1_1$).*

Setting aside the question of Turing-completeness, and concentrating exclusively on establishing $\underline{\leftrightarrow} \notin \Pi^1_1$, we can do without Friedman [23], appealing instead to Grigorieff [24] and $\Sigma^1_1$-boundedness to deduce

**Corollary 10.** *There is a transition predicate computable in DTIME-SPACE$(n, log(n))$ over which bisimilarity is not $\Pi^1_1$.*

**Proof.** By the result announced as the title of Grigorieff [24], the linear order $\to_{\hat{a}}$ above has an isomorphic copy in DTIME-SPACE$(n, log(n))$, call it $\Rightarrow$. Let $n$ be the image in $\Rightarrow$ of the top element 0 of $\to_{\hat{a}}$ (corresponding to $\hat{a}$), and note that bisimilarity $\underline{\Leftrightarrow}$ over $\Rightarrow$ cannot be $\Pi^1_1$, or else $\Rightarrow$ restricted to

$$\{m \in \omega \quad \mid \quad n \Rightarrow m \text{ and not } n \underline{\Leftrightarrow} m\}$$

is a $\Sigma^1_1$ well-ordering of length $\omega_1^{CK}$ (contradicting $\Sigma^1_1$-boundedness). ⊣

Theorem 9 and Corollary 10 provide intrinsic measures of bisimilarity's complexity, from which it follows that a logical characterization (à la Hennessy-Milner) of bisimilarity (over transitions computable in linear time) requires a non-hyperarithmetic notion of satisfaction. Reducing bisimilarity to its finitary approximations (as in Theorems A and A′) just shoves the problem over to the transition predicate (polluting it with non-standard programs); the complexity of what is analyzed by induction (or co-induction) becomes trivial compared to what is assumed at the base case! An (arithmetic) $\omega$-rule is simply insufficient to capture a non-$\Pi^1_1$ concept. A logic for bisimilarity requires more than $\Pi^1_1$ notions of inference well-known to be adequate for dynamic logic (under translations described, for example, in Harel [25]). We are led to a flight in logical complexity, not unlike that suggested in Darondeau and Yoccoz [15].

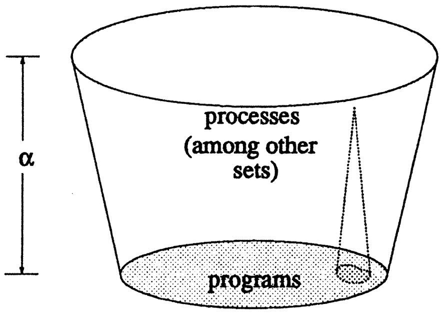
## 5 Discussion: analyzing `mechanical` transitions *abstractly*

The rather abstract investigations above may seem so far removed from the reality of mechanical computation that we might ask if one has anything to do with the other. It is true enough that in studying mechanical computation, a common attitude, when faced with matters involving astronomical complexity, is to retreat to some fragment of the logic that is decidable, and is therefore, in principle, amenable to mechanical computation. But understanding mechanical computation is different from mechanical computation, and logical abstractions introduced for the former (e.g., input/output equivalence) are bound (every now and then) to exceed the realm of mechanical computation. And when these do, we should not forget that retreating to some "mechanically tractable" approximation of these abstractions leaves the *monstrously* complex reduction of reasoning to that fragment un-analyzed. The larger mechanically uncomputable whole still hangs over us, begging for our attention, and dwarfing the part amenable to mechanical computation (however carefully we examine that fragment). And before, for instance, dismissing complications based on infinite branching as marginal, we might keep in mind that the notion of infinity exists because, in the absence of a fixed finite bound, it has proved to be a useful abstraction (functioning, as it were, as a "telescope" into the unknown).

Relating these considerations more concretely to the force of the present paper, a natural question, given that bisimilarity over mechanical processes may not even be $\Pi^1_1$,[4] is

*is bisimilarity a reasonable notion of process equality?*

So long as the concept of a process is, however, understood as a logical *abstraction* (grounded, hopefully, in a `mechanical` reality), why impose an absolute limit on the logical complexity of a notion of process equality? Abstract reasoning and mechanical computation are two very different activities. Abstract concepts are our friends, and to insist that they be mechanically computable would be to seek rather dull company. What matters (from the point of view of theoretical computer science) is that they have something interesting to say *about* mechanical computation. The basic thrust of the present work has been to suggest the notion of a bisimulation as a link grounding generalized (abstract) recursion theory in ordinary (mechanical) recursion theory, according to the picture



adapted from Barwise [7] (pp. 42, 43, etc.). The picture describes a universe of sets built hierarchically along an ordinal $\alpha$ from a collection of urelements, which (in the present case) are given by programs, construed as syntactic objects, and analyzed semantically as processes (i.e., sets[5]). The ordinal $\alpha$ is, by a theorem of Gandy's (see Barwise [7], p. 211) an upper bound on the closure ordinal for the operator $\cdot^{bf}$ co-inductively computing bisimilarity. The upper bound is, in fact, tight in that it corresponds to the next admissible ordinal, as discussed in Fernando [18], where the relevance of generalized recursion theory to operational semantics is argued and developed further.

## References

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

[2] L. Aceto, B. Bloom, and F. Vaandrager. Turning SOS rules into equations. In *Proceedings, Seventh Annual Symposium on Logic in Computer Science.* IEEE Computer Society Press, Washington, 1992.

[3] Peter Aczel. An introduction to inductive definitions. In J. Barwise, editor, *Handbook of Mathematical Logic.* North-Holland, Amsterdam, 1977.

[4]To be sure, other objections have been raised against bisimilarity — for example, that it is too fine (e.g., Bloom, Istrail and Meyer [13]) and that it may fail to be a congruence for certain transformations on states. Without claiming to pronounce the final word on the matter, let us just say that the criticism concerning fineness is difficult to take seriously if translations of transition systems are allowed (— surely input/output equivalence is not too fine ? —); and that as for bisimilarity failing to be a congruence for certain functions, the blame need not be put on bisimilarity but rather on the non-compositional functions introduced to a transition system. Afterall, it is easy enough to construct an r.e. transition relation containing copies of all r.e. transition relations (i.e., a "universal operational semantics") without having to introduce function symbols into the signature $\{\dot{L}, \leadsto\}$ of transition systems. One is then certainly free to expand that $\{\dot{L}, \leadsto\}$-model to interpret all sorts of pathological functions, but it would be too much to expect bisimilarity to respect all such functions indiscriminately, especially since the notion of a bisimulation presupposes a fixed level of abstraction (i.e., transition predicate), of which there are a multitude.

[5]Of course, the notion of a set can arise in other ways, as in the translation $\mathcal{L}$ in section 3, where they occur as states.

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

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

[6] J. Barwise, R.O. Gandy, and Y.N. Moschovakis. The next admissible set. *Journal of Symbolic Logic*, 36(1), 1971.

[7] Jon Barwise. *Admissible Sets and Structures.* Springer-Verlag, Berlin, 1975.

[8] J. van Benthem and J. Bergstra. Logic of transition systems. Technical report, ILLC, Amsterdam, March 1993.

[9] J. van Benthem, J. van Eijck, and V. Stebletsova. Modal logic, transition systems and processes. Technical Report CS-R9321, CWI, Amsterdam, March 1993.

[10] Johan van Benthem. *Modal Logic and Classical Logic.* Bibliopolis, Napoli, 1985.

[11] Johan van Benthem. *Language in action: categories, lambdas and dynamic logic.* North-Holland, Amsterdam, 1991.

[12] Johan van Benthem. Which program constructions are safe for bisimulation? Technical report, ILLC, Amsterdam, February 1993.

[13] 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 *Journal of the ACM*.

[14] S. Christensen, Y. Hirshfield, and F. Moller. Decomposability, decidability and axiomatisability for bisimulation equivalence on basic parallel processes. In *Proc. LICS 93*, To appear (1993).

[15] Ph. Darondeau and S. Yoccoz. Proof systems for infinite behaviours. *Information and Computation*, 99(2), 1992.

[16] Philippe Darondeau. Concurrency and computability. In I. Guessarian, editor, *Semantics of Systems of Concurrent Processes*, LNCS 469. Springer-Verlag, Berlin, 1990.

[17] Solomon Feferman and Clifford Spector. Incompleteness along paths in progressions of theories. *Journal of Symbolic Logic*, 27(4), 1962.

[18] Tim Fernando. A path from generalized recursion back to mechanical computation. In preparation.

[19] Tim Fernando. Transition systems and dynamic semantics. In *Logics in AI*, LNCS 633 (subseries LNAI). Springer-Verlag, Berlin, 1992. A slightly corrected version has appeared as CWI Report CS-R9217, June 1992.

[20] Tim Fernando. From partial equivalence relations to bisimulations. Manuscript, 1993.

[21] Tim Fernando. Between programs and processes: absoluteness and open ended-ness. Technical Report IAM 92-011, Institut für Informatik, Universität Bern, July 1992.

[22] Tim Fernando. Comparative transition system semantics. In *Computer Science Logic: Selected Papers from CSL '92*. Springer-Verlag, Berlin, To appear. An earlier version appeared as CWI Report CS-R9222, July 1992.

[23] Harvey Friedman. Recursiveness in $\Pi^1_1$ paths through $\mathcal{O}$. *Proc. Amer. Math. Soc.*, 54, January 1976.

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

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

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

[27] H. Jerome Keisler. Forcing and the omitting types theorem. In M. Morley, editor, *Studies in Model Theory*. The Mathematical Association of America, 1973.

[28] H. Jerome Keisler. Fundamentals of model theory. In J. Barwise, editor, *Handbook of Mathematical Logic*. North-Holland, Amsterdam, 1977.

[29] Jan Willem Klop. Lectures on bisimulation semantics. The material on 'ordinal processes' (cited above) appeared in course notes for lectures given at the REX workshop, Noordwijkerhout, May 1988, but regretably *not* in the proceedings of the workshop (LNCS 354).

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

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

[32] Gordon D. Plotkin. A structural approach to operational semantics. Technical Report DAIMI FN-19, Computer Science Department, Aarhus University, 1981.

[33] 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.