Transition Systems and Dynamic Semantics

Tim Fernando
fernando@cwi.nl

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

Abstract. Transition systems over first-order models, first-order theories, and families of first-order models are constructed and examined in relation to dynamic semantics (more specifically, DPL, DRT and Update logic). Going the other direction, first-order models are extracted from transition systems, bringing full circle the connection between static and dynamic notions. Only states computationally accessible from an initial state (with minimal information content) are considered, motivating the introduction of an internal notion of proposition on which the concept of an update is analyzed.

Key words and phrases: transition system, bisimulation, dynamic semantics, first-order logic, updates.
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'). The author is gratefully indebted to J. van Benthem, J. van Eijck, C. Gardent, M. Kracht, W. Meyer-Viol and K. Vermeulen for helpful discussions and to CWI for refuge.

As described in van Benthem [4], there is a growing interest in developing a certain broad conception of logic as the processing of information. A particularly important theme to emerge in work on natural language semantics loosely labelled "dynamic semantics" (important examples of which include Kamp [13], Heim [12], Groenendijk and Stokhof [9], and Veltman [20]) is to locate the essence of a proposition in the set of transitions between states (or contexts) that it induces. The present paper analyzes a link uncovered in Groenendijk and Stokhof [9] between these sets of transitions and programs in dynamic logic (see, for example, Harel [11]). The key notion studied is that of a "transition system", the basic thesis being that a good deal of what makes dynamic semantics "dynamic" are constructions that can be fruitfully understood against a background of "static" notions from first-order logic.

Identifying the set of transitions effected by a proposition with a program, a *transition system* is defined, relative to a collection Π of *programs*, to be a triple $\langle S, \{\xrightarrow{\pi}\}_{\pi\in\Pi}, s_0\rangle$ consisting of a non-empty set S of *states*, a family of binary relations $\xrightarrow{\pi} \subseteq S \times S$ on the states (meant as the interpretation of π), and an *initial state* $s_0 \in S$. The "dynamic" shift in perspective on propositions is analyzed below in terms of transition systems built from first-order models, families of first-order models, and first-order theories, in accordance with dynamic logic. In each case, an initial state s_0 with minimal information content is isolated, and the states considered are restricted to those accessible from s_0 by a finite sequence of programs. This simple move is shown to have rather far reaching consequences.

(i) It is exploited in the technical arguments of sections 1 and 3 (some of the details of which are relegated to the Appendix) to exhibit a certain duality between static and dynamic notions.
(ii) It forms the basis for the introduction in section 4 of an internal notion of proposition employed in an account of updates.

The reader familiar with the relevant formal systems will find that (i) relates to DPL (Groenendijk and Stokhof [9]) and DRT (Kamp [13], Heim [12]), while (ii) reconsiders Update logic (Veltman [20]), proposing, in particular, a "syntactic" treatment of might. Along the way, the growth of information between partial states is investigated (section 2), and a view of discourse interpretation as consisting, in part, of the construction of a first-order theory is developed.

1 Transition systems over first-order models

Given a first-order signature (= vocabulary = set of non-logical symbols)[1] L (with equality) and a countable set X of variables, dynamic logic associates with every first-order L-model $M = \langle |M|, \ldots \rangle$ a transition system $[\![M]\!]$ whose states map finitely many variables from X to objects in the universe $|M|$ of M. The collection of programs π involved are given by

$$\pi ::= \varphi? \mid x :=? \mid \pi_1; \pi_2 \mid \pi_1 + \pi_2 \mid \pi^* \mid \neg\pi$$

over atomic L-formulas φ with variables x from X. As it turns out, the combination of * and $\neg$ leads to various complications that are best dealt with by confining * to the Appendix, since * has as yet found no applications in the systems of dynamic semantics to be considered below. So, let Π_L be the collection of programs π *without* an occurrence of *. The initial state of $[\![M]\!]$ is the empty valuation $\emptyset$, from which the remaining states — valuations $\alpha, \beta, \ldots$ from a (non-empty) finite subset of X into $|M|$ — can be obtained through a sequential composition of "random assignments" $x :=?$. More precisely, writing $[\![\pi]\!]_M$ for the relations $\xrightarrow{\pi}$ of $[\![M]\!]$, define

$$\begin{aligned}
\alpha[\![\varphi?]\!]_M\beta &\text{ iff } \alpha = \beta \text{ and } M \models \varphi[\alpha] \\
\alpha[\![x :=?]\!]_M\beta &\text{ iff } \alpha = \beta \text{ except possibly at } x, \text{ and } x \in \text{domain}(\beta) \\
\alpha[\![\pi_1; \pi_2]\!]_M\beta &\text{ iff } \alpha[\![\pi_1]\!]_M\gamma \text{ and } \gamma[\![\pi_2]\!]_M\beta \text{ for some } \gamma \\
\alpha[\![\pi_1 + \pi_2]\!]_M\beta &\text{ iff } \alpha[\![\pi_1]\!]_M\beta \text{ or } \alpha[\![\pi_2]\!]_M\beta \\
\alpha[\![\neg\pi]\!]_M\beta &\text{ iff } \alpha = \beta \text{ and there is } \textit{no } \gamma \text{ for which } \alpha[\![\pi]\!]_M\gamma\ .
\end{aligned}$$

Note that if α is related to itself by the test $[\![\varphi?]\!]$, then α must be defined on all the variables in φ. This leads to a certain partiality in that, for instance, it is *not* the case that for some α, $\emptyset[\![x = x?]\!]_M\alpha$ (or, $\emptyset[\![x \neq x?]\!]_M\alpha$). Also, because of closure under negation $\neg\pi$ (written as such in Groenendijk and Stokhof [9], but known more

[1] Certain basic notions from model theory are taken for granted in what follows; definitions can be found in the initial sections of Keisler [16].

traditionally as the test $[\pi]\bot?$), tests for arbitrary first-order L-formulas (on X) as well as modal formulas built from programs can be defined, following

$$\begin{aligned}(\varphi\&\psi)? &= \varphi?;\psi?\\ [\pi]\,\varphi? &= \neg(\pi;\neg(\varphi?))\\ \exists x\varphi? &= \neg\neg(x :=?;\varphi?)\ .\end{aligned}$$

Our first theorem, however, depends only on the inclusion of atomic tests and random assignments among the programs.

Theorem 1. *For (finite or) countable L-models M and N, $M \cong N$ iff $[\![M]\!] \cong [\![N]\!]$.*

The key to the proof of Theorem 1 (which along with other proofs for this section can be found in the Appendix) lies in the notion of a *partial isomorphism set* from M to N (see, for example, Keisler [16]), which enables an isomorphism between M and N to be built "back and forth". This notion, in turn, corresponds to an equivalence between $[\![M]\!]$ and $[\![N]\!]$, which can be defined more generally for any two transition systems over a set Π of programs as follows. A *bisimulation* (Park [18]) between $\langle S, \{\overset{\pi}{\rightarrow}\}_{\pi\in\Pi}, s_0\rangle$ and $\langle S', \{\overset{\pi}{\rightarrow'}\}_{\pi\in\Pi}, s'_0\rangle$ is a relation $R \subseteq S \times S'$ such that whenever sRs', then for every $\pi \in \Pi$,

$$\forall t \overset{\pi}{\leftarrow} s\ \exists t' \overset{\pi}{\leftarrow'} s'\ tRt' \text{ and } \forall t' \overset{\pi}{\leftarrow'} s'\ \exists t \overset{\pi}{\leftarrow} s\ tRt'\ .$$

$\langle S, \{\overset{\pi}{\rightarrow}\}_{\pi\in\Pi}, s_0\rangle$ and $\langle S', \{\overset{\pi}{\rightarrow'}\}_{\pi\in\Pi}, s'_0\rangle$ are *bisimilar*, written

$$\langle S, \{\overset{\pi}{\rightarrow}\}_{\pi\in\Pi}, s_0\rangle \leftrightarroweq \langle S', \{\overset{\pi}{\rightarrow'}\}_{\pi\in\Pi}, s'_0\rangle\ ,$$

if there is a bisimulation between them relating s_0 to s'_0. Now, the predicate

$$[\![M]\!] \leftrightarroweq [\![N]\!]$$

can be added to the list of equivalent predicates in Theorem 1.

Theorem 1 states that no information is lost in the passage from a countable first-order model M to its transition system $[\![M]\!]$. A more abstract (i.e., information-decreasing) analysis better suited to bring out the first-order character of M is provided by the following standard construction on transition systems, applied to $[\![M]\!]$. For every $\pi \in \Pi_L$, let

$$s_{\pi,M} = \{\alpha \mid \emptyset[\![\pi]\!]_M\alpha\}\ ,$$

and

$$[\pi]_M = \{(s_{\pi',M}, s_{\pi';\pi,M}) \mid \pi' \in \Pi_L,\ s_{\pi',M} \neq \emptyset \neq s_{\pi';\pi,M}\}\ .$$

The states of the new transition system $[M]$ are the *non-empty* sets $s_{\pi,M}$ for some $\pi \in \Pi_L$, the initial state being $\{\emptyset\}$ (i.e., $s_{\neg(x:=?),M}$). The interpretation of π under $[M]$ is $[\pi]_M$, making the transition system *deterministic* in that the transition relations are partial functions.

Example. Suppose L includes two constants 0 and 1, M is an L-model interpreting 0 as 0 and 1 as 1, and $\hat{\pi}$ is the non-deterministic program $x := 0 + x := 1$ (or, to be more precise, $x :=?; (x = 0? + x = 1?)$). Then

$$\emptyset \; [\![\hat{\pi}]\!]_M \; \alpha \text{ iff } \alpha = \{(x,0)\} \text{ or } \alpha = \{(x,1)\}$$
$$\{\emptyset\} \; [\hat{\pi}]_M \; s \text{ iff } s = \{\{(x,0)\},\{(x,1)\}\} \; .$$

Note also that $s_{\pi_0,M} \; [\pi]_M \; s_{\pi_1,M}$ does *not* imply that $\pi_1 = \pi_0;\pi$ (although the converse holds). Take π_0 to be $\neg(x :=?)$, π_1 to be $y :=?$, and π to be $(x = 0)?+y :=?$.

The next theorem should be contrasted with Theorem 1, recalling that over infinite first-order models, isomorphism is typically a much stronger property than elementary equivalence $\equiv_L$ (as a consequence, for instance, of compactness).

Theorem 2. *For L-models M and N, the following are equivalent.*

1. $M \equiv_L N$.
2. $[M] \cong [N]$.
3. $[M] \leftrightarrows [N]$.

Theorem 2 suggests that a transition system be built directly from a first-order theory, rather than a single first-order model. Before proceeding on to such constructions, we pause to consider how information grows between partial states.

2 Partiality of states and information growth

Intuitively, the initial states $\emptyset$ and $\{\emptyset\}$ of $[\![M]\!]$ and $[M]$ respectively have *no* information content — at least when compared with the other states. More formally, $\emptyset$ is the least $[\![M]\!]$-state under the subfunction partial order $\subseteq$, whereas $\{\emptyset\}$ is a minimal $[M]$-state under the so-called "Smyth pre-order" $\leq$ with respect to $\subseteq$

$$s \leq s' \text{ iff } \forall \beta \in s' \; \exists \alpha \in s \; \alpha \subseteq \beta \; .$$

The partial order $\subseteq$ on $[\![M]\!]$-states exposes the "expansive" character of random assignments which more complicated programs can inherit.[2] The move from finite valuations in $[\![M]\!]$ to accessible sets of finite valuations in $[M]$ captures the "eliminative" character of tests $\varphi?$ without discarding the partial order on finite functions. Thus, a program can increase information two ways — expansively and eliminatively. This is not to say, however, that information can never be lost after the execution of a program.

[2] The idea of treating states as partial objects is, of course, hardly novel, going back at least to Goldblatt [8]. A particularly rich structure is introduced in Vermeulen [21], where states range over sequences (recording the values assigned to variables), and an explicit "downdating" program construct is added to provide the possibility of destroying information. The present paper is concerned with a more primitive notion of program variable.

Applied to an $[\![M]\!]$-state already defined on x, the random assignment $x :=?$ destroys information. To protect information while allowing for the possibility of its expansive growth, it is convenient to introduce a *guarded assignment*

$$x := *$$

defined as

$$x = x? + \neg(x = x?); x :=? ,$$

the idea being to assign a value to x iff no value has so far been assigned to it. Guarded assignments are examples of *monotone* programs — i.e., programs π such that for every L-model M and $[\![M]\!]$-states α, β $\alpha[\![\pi]\!]_M\beta$ implies $\alpha \subseteq \beta$. Note that every $\pi \in \Pi_L$ can be made monotone by guarding random assignments.

Turning to the second dimension of information growth, call a program π *eliminative* if whenever $\alpha[\![\pi]\!]_M\beta$, it follows that $\alpha = \beta$; whence, for such π, $s[\pi]_M s'$ implies $s' \subseteq s$. Note that tests are eliminative, whereas random (or guarded) assignments are not. Eliminative programs are those programs that can be characterized (statically) by propositions. The point is that every program π has an eliminative approximation $\neg\neg\pi$ that can be constructed from what van Benthem [4] calls *modes* from (static) propositions to (dynamic) procedures, and *projections* going the opposite direction. Assuming a notion of proposition closed under program-labelled modalities (i.e., $\langle\pi\rangle$ and $[\pi]$), compose the projection $\pi \mapsto \langle\pi\rangle\top$ (where $\top$ is, say, $\exists x\ x = x$) with the mode $\varphi \mapsto \varphi?$, where

$$\alpha[\![\varphi?]\!]_M\beta \text{ iff } \alpha = \beta \text{ and } M \models \varphi[\alpha] .$$

This yields the map $\pi \mapsto \langle\pi\rangle\top?$, where

$$\begin{aligned}\alpha[\![\langle\pi\rangle\top?]\!]_M\beta &\text{ iff } \alpha = \beta \text{ and there } \textit{is} \text{ a } \gamma \text{ such that } \alpha[\![\pi]\!]_M\gamma \\ &\text{iff } \alpha[\![\neg\neg\pi]\!]_M\beta .\end{aligned}$$

Observe that $\langle\pi\rangle\top?$ (i.e., $\neg\neg\pi$) is semantically equal to π if π is eliminative, while $\langle\varphi?\rangle\top$ is logically equivalent to φ.

3 From semantic evaluation to semantic construction

Two important formalisms associated with dynamic semantics are DPL (Groenendijk and Stokhof [9]) and DRT (Kamp [13], Heim [12]). For our present purposes, it suffices to describe DPL roughly (and a bit incorrectly) as a subsystem of (predicate) dynamic logic subsumed by the *-free programs of section 1, and DRT as a formalism for analyzing the processing of natural language discourse. Hence, whereas the transition systems of section 1 might be associated with the former, it is more natural, for the latter case, to build transition systems out of more partial information — partial information that might be embodied by families of first-order models, or by possibly incomplete theories —, and to extract first-order models from such transition systems. These are presently taken up, in turn.

3.1 Deterministic transition systems induced partially

Given a family $\mathcal{M}$ of first-order L-models, form the transition system $[\mathcal{M}]$ from the transition systems $[M]$, for $M \in \mathcal{M}$, as follows. The states of $[\mathcal{M}]$ are non-empty sets of the form

$$\{(M, s_{\pi,M}) \mid M \in \mathcal{M} \text{ and } s_{\pi,M} \neq \emptyset\}$$

for $\pi \in \Pi_L$. Call the set above, provided it is non-empty, $s_{\pi,\mathcal{M}}$. The initial $[\mathcal{M}]$-state is $s_{\hat{\pi},\mathcal{M}}$ where $\hat{\pi}$ is $\neg x :=?$. The interpretation $[\pi]_{\mathcal{M}}$ of π under $[\mathcal{M}]$ is defined by

$$s_{\pi_0,\mathcal{M}} \; [\pi]_{\mathcal{M}} \; s_{\pi_1,\mathcal{M}} \text{ iff } s_{\pi_1,\mathcal{M}} = s_{\pi_0;\pi,\mathcal{M}} \; .$$

By Theorem 2,

(†) if $\forall M \in \mathcal{M} \; \exists M' \in \mathcal{M}' \; M \equiv_L M'$ and $\forall M' \in \mathcal{M}' \; \exists M \in \mathcal{M} \; M \equiv_L M'$ then $[\mathcal{M}] \leftrightarrows [\mathcal{M}']$.

The converse of (†), however, fails, a counter-example to which will be supplied after presenting an alternative characterization of $[\mathcal{M}]$ in terms of the set $Th\mathcal{M}$ of all L-sentences true in every model of $\mathcal{M}$.

Given a consistent L-theory T, define the following transition system $[T]$. Let s_π be

$$\{(X_0, \psi) \mid X_0 \text{ is the set of variables} \in X \text{ occurring in } \pi, \text{ and}$$
$$\text{for every } L\text{-model } N \text{ and } \beta : X_0 \to |N|, \; \emptyset[\![\pi]\!]_N\beta \text{ iff } N \models \psi[\beta]\}.$$

(It is possible to define s_π without referring to, or quantifying over, L-models N; see Lemma A$^\neg$ of the Appendix.) $[T]$-states are non-empty sets of the form

$$\{(X_0, \psi) \in s_\pi \mid \psi \text{ is consistent with } T\} \; ,$$

for $\pi \in \Pi_L$. Call the set above, provided it is non-empty, $s_{\pi,T}$, and let the initial $[T]$-state $s_{\hat{\pi},T}$ be that induced by $\hat{\pi} = \neg(x :=?)$. The interpretation $[\pi]_T$ of π under $[T]$ is defined by

$$s_{\pi_0,T} \; [\pi]_T \; s_{\pi_1,T} \text{ iff } s_{\pi_1,T} = s_{\pi_0;\pi,T} \; .$$

The definition above was formulated carefully in order to accomodate the possibility of non-monotonicity (i.e., revision in X) as well as partiality (again, due to X). (Readers familiar with DRT might compare $[T]$ with DRS's.) Without requiring any essentially new ideas for its proof,[3] Theorem 2 admits the following generalization.

Theorem 3. *For families $\mathcal{M}$ and $\mathcal{M}'$ of L-models, the following are equivalent.*

1. $Th\mathcal{M} = Th\mathcal{M}'$.
2. $[Th\mathcal{M}] \cong [\mathcal{M}']$.
3. $[\mathcal{M}] \cong [\mathcal{M}']$.
4. $[\mathcal{M}] \leftrightarrows [\mathcal{M}']$.

[3] More specifically, appeal first to Lemma A$^\neg$ to prove $[Th\mathcal{M}] \cong [\mathcal{M}]$, and then show, as in Proposition B$^\neg$, that the only bisimulation on $[\mathcal{M}]$ is equality.

5. $[Th\mathcal{M}] \underline{\leftrightarrow} [\mathcal{M}']$.

Returning now to (†), a counter-example is provided by taking L to consist of a single binary relation, $\mathcal{M}$ to be the family of all finite linear orders, and $\mathcal{M}'$ to be $\mathcal{M}$ together with an infinite linear order constructed, by compactness, to satisfy $Th\mathcal{M}$.

3.2 First-order models from transition systems

To complete the duality between static notions (first-order models, L-formulas φ, and $\equiv_L$) and dynamic notions (transition systems, programs π, and $\underline{\leftrightarrow}$), first-order models are extracted from transition systems below. Thus, transition systems are shown to serve a dual role: a semantic one related to truth (building *on* first-order models), and also a constructive one (*building* first-order models).

Recall (or consult, for example, Keisler [16]) that the usual proof for the completeness theorem of first-order logic consists of the steps

(1) expand the language with (Henkin) witnesses, and
(2) complete a consistent Henkin theory (Lindenbaum's theorem),

followed by a quotient construction over constants. The first step corresponds to the expansive growth of information in a state, and the second, to eliminative growth.

Proposition 4. *There is a sequence $\{\pi_i\}_{i<\omega}$ of monotone programs such that for every L-theory T and countable L-model N, the following are equivalent*

1. *There is a map f from X onto $|N|$ such that for every $n < \omega$, there is a $(X_0, \psi) \in s_{\pi_0;\ldots;\pi_n,T}$ for which ψ is true in N, relative to f.*
2. *N is a model of T.*

Furthermore, given a countable L-model M, there is a sequence $\{\pi_i^M\}_{i<\omega}$ of monotone programs such that for every L-model N, the following are equivalent.

1'. *There is a map f from X onto $|N|$ such that for every $n < \omega$, there is an $\alpha \subset f$ such that $\emptyset[\![\pi_0^M; \ldots; \pi_n^M]\!]_N \alpha$.*
2'. *$N \cong M$.*

Proof. Fix a well-ordering on X, and an enumeration $\{\psi_i\}_{i\in\omega}$ of all L-formulas with variables from X. For every such L-formula φ, let π_φ be

$$x_0 := *; \ldots; x_k := *; \varphi?$$

where $x_0, \ldots, x_k$ are φ's free variables (enumerated according to the fixed ordering). For every $i < \omega$, define a program $\hat{\pi}_i$ inductively as follows.

> Let x be the least variable in X not occuring in $\hat{\pi}_j$ for $j < i$. If $\psi_i = \exists y \psi$, then for the least variable x' different from x or any variable occuring in ψ or $\hat{\pi}_j$ for $j < i$, define $\hat{\pi}_i$ to be
>
> $$x := * \ ; \ \pi_{\psi[x'/y]} \ .$$
>
> Otherwise (i.e., ψ_i is not existential), let $\hat{\pi}_i$ be
>
> $$x := * \ ; \ \pi_{\psi_i} \ .$$

(The assignments $x := *$ are inserted so that $|N| = \{f(x) \mid x \in X\}$.) Now, for $i < \omega$, let π_i be the program

$$\hat{\pi}_i + \neg\hat{\pi}_i \ ,$$

which not only decides the truth of ψ_i but also specifies a witness if ψ_i is existential and tests successfully. That is, running the programs π_i (in sequence $i = 0, 1, \ldots$) over a countable L-model N yields (at the limit) a consistent, complete theory with witness set X, where each element of $|N|$ is the value of some variable in X.

Constructing $\{\pi_i^M\}_{i\in\omega}$ is similar, except that it is carried out relative to an enumeration $\{m_i\}_{i\in\omega}$ of $|M|$ so that the choice between $\hat{\pi}_i$ and $\neg\hat{\pi}_i$ is determined instead by the truth or falsehood of ψ_i relative to the finite part of f built at stage i. $\dashv$

4 Updates and an internal notion of proposition

As an analysis of the notion of an update at the propositional level, Veltman [20] presents the following system of Update logic, denoted U below. Fix a set $\mathcal{A}$ of propositional variables (written $p, \ldots$). A *world* w is a subset of $\mathcal{A}$, and an *information state* σ is a set of worlds. An *update* A is generated according to

$$A ::= \ p \mid \ \sim A \mid A \wedge B \mid A \vee B \mid \mathsf{might}\ A \mid A; B$$

and induces a total function on information states as follows

$$\begin{aligned}
\sigma[p] &= \{w \in \sigma \mid p \in w\} \quad \text{for } p \in \mathcal{A} && (1)\\
\sigma[\sim A] &= \sigma - \sigma[A] && (2)\\
\sigma[A \wedge B] &= \sigma[A] \cap \sigma[B] \\
\sigma[A \vee B] &= \sigma[A] \cup \sigma[B] \\
\sigma[\mathsf{might}\ A] &= \sigma \quad \text{if } \sigma[A] \neq \emptyset && (3)\\
&= \emptyset \quad \text{otherwise} && (4)\\
\sigma[A; B] &= (\sigma[A])[B] \ .
\end{aligned}$$

Note that for every σ and A, $\sigma[A] \subseteq \sigma$, and that $\emptyset$ is an "absurd" information state from which there is no chance of recovery. Hence, instead of interpreting updates as total functions on a set of information states that includes an absurd one, updates can be taken to be partial functions on a set of non-empty (i.e., non-absurd) information states. From this point of view, might A is "static" in that it can only relate identical states, whereas other programs may relate distinct states.

But what is the intuition behind the notion of an information state above? An information state is a set of worlds, which in turn can be regarded as functions from $\mathcal{A}$ to a set of two truth values. In fact, *total* valuations — in view of clauses (1) and (2) and the fact that the powerset $2^{\mathcal{A}}$ of $\mathcal{A}$ corresponds to the function space from $\mathcal{A}$ to a two-element set. Observe that the defined transitions are "static" in that the worlds in the states do not change: they either persist unaltered through the transition or drop out. Ignoring the might-clauses (3) and (4) for the moment,

the updates can be understood as perfectly ordinary eliminative tests, assuming, that is, that the worlds they act on are total. Take away this assumption — i.e., make the worlds (= valuations) finite —, and the question arises as to how an update behaves on a partial world [sic] in which the truth of the proposition (being updated) is undecided. It is important to address this question because, from the point of view of computational practice, total valuations certainly have *no* priority over finite ones.[4] An account based on total valuations must demonstrate that it faithfully models the reality of finite valuations. And as far as the reality of finite valuations is concerned, an update must not only be able to test, but (in the absence of information about the proposition being updated) also stipulate (i.e., establish). That is, we are led to a different conception of updates, which, as we will however see, strips away some of the mystery in might.

4.1 A "dynamic" reconstruction of propositional updates

An account of U within the framework of section 1 can be given based on the Boolean algebra 2 over $\{0,1\}$, with $L = \{0,1\}$ and the set X of variables equal to the set $\mathcal{A}$ of propositional variables of U. A map $\cdot^u$ from updates A to programs $A^u \in \Pi_L$ is defined as follows. For $p \in \mathcal{A}$, the corresponding program p^u must test if p is true, asserting it to be the case if its truth has not been previously determined. That is, define

$$p^u = p := * \ ; \ p = 1? \ .$$

Extend the translation to might by defining

$$(\text{might } A)^u = \neg\neg(A^u) \quad (= \langle A^u\rangle\top?) \ ,$$

which is to say that $(\text{might } A)^u$ checks that a transition through A^u is possible, without actually making such a transition.[5] In particular, $(\text{might } p)^u$ does not assert that p is true, only that p has not been determined to be false. Next, rather than setting $(A \vee B)^u$ to $(\langle A^u\rangle\top \ \vee \ \langle B^u\rangle\top)?$ (which yields a static update), define

$$(A \vee B)^u = A^u + B^u \ ,$$

and, to preserve the commutativity of $\wedge$, let

$$(A \wedge B)^u = A^u; B^u \ + \ B^u; A^u$$

[4] Total valuations arise as convenient abstractions from finite valuations. The idea of modelling a finite function as the set of its total extensions not only tends to confuse eliminative with expansive information growth, but also replaces a perfectly finite object with one that is two times infinite: an infinite set of infinite objects. It is perhaps naive to expect that the foundational complications involved in this "reduction" will forever remain buried, even as extensions to richer contexts (for example, predicate formalisms) are considered. At the propositional level, however, it is quite understandable that the expansive growth of information should be overlooked, given the absence of the notion of a program assignment.

[5] Recall that $\neg$ applies to programs, and should be distinguished from the negation $\sim$ in (2) that operates on propositions.

instead of simply $A^u; B^u$, which we reserve for $(A; B)^u$

$$(A; B)^u = A^u; B^u \ .$$

As for $(\sim A)^u$, a treatment of falsehood F symmetric with truth T suggests defining (simultaneously with the above) the "negative" translation A_u as follows

$$\begin{aligned} p_u = p := * \ ; \ p = 0? \qquad & \text{for } p \in \mathcal{A} \\ (A \wedge B)_u &= A_u + B_u \\ (A \vee B)_u &= A_u; B_u \ + \ B_u; A_u \ , \end{aligned}$$

the point being to take

$$\begin{aligned} (\sim A)_u &= A^u \\ (\sim A)^u &= A_u \ . \end{aligned}$$

The laws of double negation and de Morgan then hold (as in U)

$$\begin{aligned} (\sim\sim A)^u &= A^u \\ (\sim(A \wedge B))^u &= (\sim A \vee \sim B)^u \\ (\sim(A \vee B))^u &= (\sim A \wedge \sim B)^u \ . \end{aligned}$$

For completeness, set

$$(\mathsf{might}\ A)_u = \neg(A^u) \quad (= \neg((\mathsf{might}\ A)^u)) \tag{5}$$

$$(A; B)_u = A_u + (\mathsf{might}\ A)^u; B_u \ . \tag{6}$$

Equation (6) respects the sequential order in ";", as well as a principle of minimal change exemplified in the definition above of $(A \vee B)^u$ as $A^u + B^u$, rather than $A^u + B^u + A^u; B^u + B^u; A^u$. The treatment by (5) of might is static, although it is plausible to equate $(\mathsf{might}\ A)_u$ with the dynamic assertion A_u. That is, the update "it is not the case that it might rain" can be construed not only as a test but as possibly establishing the truth of "it does not rain".[6] This would, however, run counter to the fact (in U) that $2^{\mathcal{A}}[\sim(\mathsf{might}\ p)] = \emptyset$. As defined, the translation $\cdot^u$ is faithful to U in a sense to be described shortly. Note that implicit in (5) is a dual must for might given by

$$\begin{aligned} (\mathsf{must}\ A)^u &= \neg(A_u) \\ (\mathsf{must}\ A)_u &= \neg\neg(A_u) \ . \end{aligned}$$

Let Π^u be the subset $\{A^u \mid A \text{ is an update}\}$ of Π_L, and, viewing 2 as an L-model, form, relative to Π^u, the transition systems $[\![2]\!]$ and $[2]$ described in section 1. (So, $[\![A^u]\!]_2$ is the interpretation under $[\![2]\!]$ of A^u; $[2]$-states are non-empty sets of finite valuations from $\mathcal{A}$ into $\{0,1\}$ accessible from $\{\emptyset\}$ by some A^u; and $[A^u]_2$ is

[6] Indeed, replacing the Boolean algebra 2 by a structure with at least 3 elements (see section 4.2), K. Vermeluen has suggested a dynamic definition of $(\mathsf{might}\ A)^u$ analogous to A^u and A_u, but neither testing that A is true nor testing that A is false.

the interpretation under [2] of A^u.) Also, associate with U the transition system U relative to Π^u whose state set is

$$\{2^{\mathcal{A}}\} \cup \{\sigma \mid \sigma \neq \emptyset \text{ and } 2^{\mathcal{A}}[A] = \sigma \text{ for some update } A\} ,$$

and whose initial state is $2^{\mathcal{A}}$. (Recall that $2^{\mathcal{A}}$ is the information state with the least information content.) Define the interpretation $\xrightarrow{A}_U$ of A^u in U by

$$\sigma \xrightarrow{A}_U \sigma' \text{ iff } \sigma[A] = \sigma'$$

for all U-states σ and σ'. While an isomorphism between [2] and U is out of the question — contrast the effects of the update $p \vee \sim p$ on $\{\emptyset\}$ and $2^{\mathcal{A}}$ —, a bisimulation is not. Towards this end, associate with every $[\![2]\!]$-state α the information state $\sigma_\alpha = \{w \mid \alpha \subset w\}$, where a world w (in the sense of U) is identified with its characteristic function from $\mathcal{A}$ to 2.

Proposition 5. *For every update A and every $[\![2]\!]$-state α, there is a finite number of $[\![2]\!]$-states $\beta_1, \ldots, \beta_n$ such that for every $[\![2]\!]$-state β and every world w,*

$$\alpha [\![A^u]\!]_2 \beta \text{ iff } \beta = \beta_i \text{ for some } i \in \{1, \ldots, n\}$$
$$w \in \sigma_\alpha[A] \text{ iff } w \supseteq \beta_i \text{ for some } i \in \{1, \ldots, n\} .$$

Proof. Proceed by induction on A simultaneously with a version for A_u (i.e., $(\sim A)^u$). ⊣

Observe that $\emptyset [\![A^u]\!]_2 \alpha$ does *not* follow from $\forall w \supseteq \alpha \; w \in 2^{\mathcal{A}}[A]$; a counter-example is provided by $A = p \vee \sim p$. On the other hand, Proposition 5 implies

Corollary 6. *For every $[\![2]\!]$-state α, and every update A,*

$$\exists \beta \; \alpha [\![A^u]\!]_2 \beta \text{ iff } \sigma_\alpha[A] \neq \emptyset .$$

In particular, the Π^u-transition systems [2] *and U are bisimilar — i.e., $[2] \leftrightarrows U$ via*

$$\{(\{\emptyset\}, 2^{\mathcal{A}})\} \cup \{(s, \sigma) \mid \exists \text{ update } A \text{ s.t. } \{\emptyset\}[A^u]_2 s \text{ and } 2^{\mathcal{A}} \xrightarrow{A}_U \sigma\} .$$

A distinction is drawn in van Eijck and de Vries [5] between an update A being "acceptable" and "accepted" in an information state σ. A is *acceptable in* σ if $\sigma[A] \neq \emptyset$ and *accepted* if $\sigma[A] = \sigma$. Corollary 6 relates directly to the former notion. As for the latter, the possibility that a program A^u is not eliminative suggests weakening the relation of equality (used in the definition of "accepted") to the so-called "Egli-Milner pre-order" $\sqsubseteq$ on the subset relation $\subseteq$

$$s \sqsubseteq s' \text{ iff } \forall \alpha \in s \; \exists \beta \in s' \; \alpha \subseteq \beta \quad \wedge \quad \forall \beta \in s' \; \exists \alpha \in s \; \alpha \subseteq \beta .$$

4.2 Prospects: from truth values to propositions

The analysis of propositions above can be generalized to structures other than the Boolean algebra 2, working with a language L consisting of the unary relation symbols T, F and D (for truth, falsehood and determinateness of truth, respectively). The idea is to replace $p = 1$ by $\mathsf{T}(p)$, $p = 0$ by $\mathsf{F}(p)$, and $p = p$ by $\mathsf{D}(p)$, redefining $p := *$ to

$$\mathsf{D}(p)? + \neg(\mathsf{D}(p)?); p :=? \ .$$

The L-theory with the axiom

$$\mathsf{T}(x) \vee \mathsf{F}(x) \supset \mathsf{D}(x)$$

can, according to the reader's taste, be expanded by various symbols, and extended by suitable axioms. For example, constants 0 and 1, a unary function symbol $\dot{\sim}$, and binary function symbols $\dot{\wedge}, \dot{\vee}$ might be added together with the axioms

$$\begin{array}{ll} \mathsf{T}(\dot{\sim}x) \equiv \mathsf{F}(x) & \mathsf{F}(\dot{\sim}x) \equiv \mathsf{T}(x) \\ \mathsf{T}(x \dot{\vee} y) \equiv \mathsf{T}(x) \vee \mathsf{T}(y) & \mathsf{F}(x \dot{\vee} y) \equiv \mathsf{F}(x) \wedge \mathsf{F}(y) \\ \mathsf{T}(x \dot{\wedge} y) \equiv \mathsf{T}(x) \wedge \mathsf{T}(y) & \mathsf{F}(x \dot{\wedge} y) \equiv \mathsf{F}(x) \vee \mathsf{F}(y) \\ \mathsf{T}(1) & \dot{\sim}0 = 1 \end{array}$$

Rather than adding the axiom $x = 0 \vee x = 1$ for bivalence, however, the move that is being suggested here is one from internalizing truth values in the L-model M to internalizing propositions. Indeed, the rudimentary L-theory above must, for the predicate case, be enriched to provide an account of the formation of a propositional object from a relation plus an appropriate tuple of arguments, where a relation need not be identified with the set of tuples satisfying it. (Note that in this case, propositions would live along side other objects in the first-order model. In other words, DPL and Update logic can be accomodated within the *-free fragment of dynamic logic, provided the language L is expanded to describe an internal notion of propositions.) There is something unmistakably intensional (or syntactic) about these internal propositions. Furthermore, their introduction into the object level is somewhat disturbing, given that an external notion of proposition is already at hand (at the meta-level). These two notions cannot, in general, be expected to coincide, in view of Tarski's theorem on the undefinability of truth (see also Montague [17], Aczel [1] and Turner [19]). The question arises as to which notion of proposition to use for analyzing updates. Appealing again to the finiteness in the computational picture underlying dynamic logic, the author is inclined to choose the internal notion not only because, being internal, it is more managable but also because the external notion (given by a first-order model) is total and, furthermore, extensional. Various reasons have been given (see, for instance, chapter 4 of Barwise [2]) to resist the identification of a proposition with the collection of (total) worlds in which it is true.[7] A somewhat persuasive case against the external notion of proposition may

[7] Returning to the matter of the undefinability of truth, an argument against the totality of worlds can be based on the Liar's paradox, as analyzed in Barwise and Etchemendy [3], and studied from a "dynamic" perspective in Groeneveld [10].

also be based on the logical intractability of a predicate version of U, in which a world is a model of some first-order theory T. In terms of transition systems, the idea is that U, lifted to first-order logic and relativized to a first-order theory T, is essentially the transition system $[T]$ described in section 3.1 above (and identified in Theorem 3 with $[\mathcal{M}]$ for a set $\mathcal{M}$ of models whose common theory is T), extended with programs might φ so that for the initial $[T]$-state s_0,

$$s_0 \ [\mathsf{might}\ \varphi]_T \ s \text{ iff } s = s_0 \text{ and } \exists s' \ s_0 \ [\varphi?]_T \ s'$$
$$\text{iff } s = s_0 \text{ and } \varphi \text{ is consistent with } T \ .$$

The problem is that most interesting first-order theories T are undecidable, whence, in contrast to the propositional case U, the predicate "φ is consistent with T" is, in general, not r.e. (in φ). Hence, neither the accepted nor the acceptable updates of this predicate system are axiomatizable.

5 Discussion

The duality between static and dynamic notions established above rests on a careful analysis of the growth of information between partial states. (Observe, for instance, that all of Theorems 1, 2, and 3, Propositions 4 and 5, and Corollary 6 use the finiteness of valuations heavily.) This is not merely an incidental feature of the present paper, but is its fundamental premise: *the computational processes necessitated by the partial character of information are central to the dynamic conception of logic* described in van Benthem [4]. (The reader is asked not to fret over the restrictions to the dynamic conception that the term "computation" might suggest, to construe the term broadly if wished, but to keep in mind that logically tractable conceptions are usually based on some coherent notion of construction.) Thus, as tempting as it may be, for instance, to "simplify matters" by passing from the finite valuations α of $[\![M]\!]$ to the total valuations used in DPL (as well as Harel [11]), the claim that the choice between finite and total valuations is immaterial to *the logic* involved is not only, under a trivial reading, false (— compare the effect of the test $x = x?$ on finite valuations versus total valuations —)[8], but assumes an understanding of "the logic" that we simply do not have, while ignoring a feature of reality (i.e., finiteness) to which we are bound. The distinction between the finite and the infinite is fundamental to foundational studies; and as formalisms are extended, the foundations on which our ideas lie must bear greater weight. To what extent the semantic picture

[8] It is true enough that programs might be restricted to those of the form $\overline{\pi}$, where $\overline{\pi}$ is π preceded by guarded assignments to variables in X occuring in π. (For example, $\overline{x =?}$ is $x := *; x = x?$.) With respect to such programs, transition systems built from finite valuations are, the author expects, bisimilar to corresponding transition systems based on total valuations (generalizing Corollary 6). It would remain then to show that the map from π to $\overline{\pi}$ provides an interpretation of the "logic" of total valuations within the "logic" of finite ones. (Similarly, replacing random assignments by guarded ones is a plausible first step towards an interpretation of first-order intuitionistic logic within the formalism of section 1.) This interpretation would reinforce the feeling (suggested already by the reality of mechanical computation) that the latter is foundationally prior, but *so what?* In reply, the reader is referred back to section 4.

of update logic can, for instance, be supported or developed in its predicate form is called to question in section 4. Looking at the matter theoretically, the expansive growth of information exposed by finite valuations allows the range of possibilities reduced by eliminative programs to be widened, while clearly staying within the scope of first-order logic, the formulas of which have an arbitrary finite number of variables.

A notable omission in the analysis above concerns the concept of inference, typically codified in a logical formalism. An analysis of a particular notion of inference presupposes a certain minimal familiarity with the underlying semantic structures. Just what this "minimal familiarity" is is a somewhat *personal* matter that is rather difficult to spell out, and should perhaps not be imposed universally. Suffice it to say that the work above was carried out with a view towards acquiring (for the author) just that familiarity, the assumption being that transition systems grounded in computation are "the underlying semantic structures." A particular outcome of this work that the author plans to develop and study further is the translation from propositions to programs described in section 4. How the various notions of inference surveyed in van Benthem [4] look under this translation and how the logical formalisms worked out in van Eijck and de Vries [6], [5] can be adapted for this purpose are two natural questions to consider. Intuitively, what is different about this translation from, say, Groenendijk and Stokhof [9], and Veltman [20] is that the program to which a proposition A translates can do one of two things: test that A is true (semantic evaluation), or establish that A is true (semantic construction). (This dual function is not a new idea, but occurs in the analysis of discourse referents in, for instance, Karttunen [14] and Heim [12].) A closer study of the transition systems $[T]$ and $[\mathcal{M}]$ restricted to the translations of propositions may be worthwhile, especially in relation to DRT and Update logic.

Lastly, notice that a basic limitation of the approaches above is that the states considered are "static." Visser [22] disputes the idea that "a state is something static", suggesting instead that "states find their natural home within the saturated-unsaturated distinction" (p. 4). Indeed, the notions of a transition system and a bisimulation are commonly applied in computer science to states that are decidedly dynamic. The interested reader is referred to Fernando [7] for relations between that tradition and dynamic logic (the basis of dynamic semantics, as understood above).

References

1. Peter Aczel. Frege structures and the notions of proposition, truth and set. In *The Kleene symposium.* North-Holland, Amsterdam, 1980.
2. Jon Barwise. *The situation in logic.* CSLI Lecture Notes Number 17, Stanford, 1989.
3. Jon Barwise and John Etchemendy. *The liar: an essay on truth and circularity.* Oxford University Press, Oxford, 1987.
4. Johan van Benthem. Logic and the flow of information. In *Proc. 9th International Congress of Logic, Methodology and Philosophy of Science.* North-Holland, Amsterdam, to appear.
5. J. van Eijck and F.J. de Vries. A sound and complete calculus for update logic. Technical Report CS-R9155, Centre for Mathematics and Computer Science, 1991.

6. J. van Eijck and F.J. de Vries. Dynamic interpretation and Hoare deduction. *Journal of Logic, Language and Information*, 1, 1992.
7. Tim Fernando. Comparative transition system semantics. Manuscript, 1992.
8. Robert Goldblatt. *Axiomatising the logic of computer programming.* LNCS 130. Springer-Verlag, Berlin, 1982.
9. J. Groenendijk and M. Stokhof. Dynamic predicate logic. *Linguistics and Philosophy*, 14, 1991.
10. Willem Groeneveld. Dynamic semantics and circular propositions. University of Amsterdam, ITLI Prepublication Series, LP-91-03, 1991.
11. David Harel. Dynamic logic. In Gabbay et al, editor, *Handbook of Philosophical Logic, Volume 2.* D. Reidel, 1984.
12. Irene Heim. The semantics of definite and indefinite noun phrases. Dissertation, University of Massachusetts, Amherst, 1982.
13. J.A.W. Kamp. A theory of truth and semantic representation. In *Formal methods in the study of language.* Mathematical Centre Tracts 135, Amsterdam, 1981.
14. Lauri Karttunen. Discourse referents. In J. McCawley, editor, *Notes from the Linguistic Underground, Syntax and Semantics 7.* Academic Press, New York, 1976.
15. H. Jerome Keisler. Forcing and the omitting types theorem. In M. Morley, editor, *Studies in model theory.* The Mathematical Association of America, 1973.
16. H. Jerome Keisler. Fundamentals of model theory. In J. Barwise, editor, *Handbook of Mathematical Logic.* North-Holland, Amsterdam, 1977.
17. Richard Montague. Syntactical treatments of modality, with corollaries on reflexion principles and finite axiomatizability. *Acta Phil. Fennica*, 16, 1963.
18. David Park. Concurrency and automata on infinite sequences. In P. Deussen, editor, *Proc. 5th GI Conference*, LNCS 104. Springer-Verlag, Berlin, 1981.
19. Raymond Turner. *Truth and Modality for Knowledge Representation.* Pitman, London, 1990.
20. F. Veltman. Defaults in update semantics. In H. Kamp, editor, *Conditionals, Defaults and Belief Revision.* Edinburgh, Dyana deliverable R2.5.A, 1990.
21. C.F.M. Vermeulen. Sequence semantics for dynamic logic. Technical report, Philosophy Department, Utrecht, 1991.
22. Albert Visser. Actions under presuppositions. Technical report, Philosophy Department, Utrecht, 1992.

Appendix

This appendix presents the technical details behind section 1 (and a bit more), building on the notation introduced there.

Theorem 1. *For countable L-models M and N, the following are equivalent.*

1. $M \cong N$.
2. $[\![M]\!] \cong [\![N]\!]$.
3. $[\![M]\!] \leftrightarrows [\![N]\!]$.

Proof. Only $3 \Rightarrow 1$ is not clear. But the definition of $[\![\pi]\!]_M$ has been arranged so that a bisimulation between $[\![M]\!]$ and $[\![N]\!]$ is a partial isomorphism from M to N (see, for example, Keisler [16]): $\emptyset R \emptyset$, and whenever $\alpha R \beta$,

(i) $\mathrm{domain}(\alpha) = \mathrm{domain}(\beta)$ (using tests "$x = x$?"),

(ii) (M, α) and (N, β) satisfy the same atomic formulas (using tests for all atomic L-formulas on X), and
(iii) for $x \in X - \text{domain}(\alpha)$,

$$\forall m \in |M| \; \exists n \in |N| \;\; \alpha \cup \{(x, m)\} \; R \; \beta \cup \{(x, n)\}$$
$$\text{and } \forall n \in |N| \; \exists m \in |M| \;\; \alpha \cup \{(x, m)\} \; R \; \beta \cup \{(x, n)\}$$

(using random assignments $x :=?$).

Thus, an isomorphism can be built "back and forth" between enumerations of $|M|$ and $|N|$. $\dashv$

Given an L-model M, it is natural to ask whether $[\![M]\!]$ is a "reduced" transition system representation of M in the sense that the only bisimulation on $[\![M]\!]$ is equality. This is the case for L-models M all of whose elements are definable (e.g., $\langle \{0, 1, \ldots\}, +, \times, 0, 1 \rangle$). On the other hand, countable counter-examples are also easy to find; take M, for instance, to be the rationals under their usual ordering. Then for any $x \in X$, any two $[\![M]\!]$-states with (the same) domain $\{x\}$ are related by a bisimulation. In particular, it follows from Theorem 1 that (for this instance of M) there is no L-model N such that $[\![N]\!]$ is the reduced form of $[\![M]\!]$.

Before turning to Theorem 2 and *-free programs, let us note that * has a certain computational basis that $\neg$ lacks, and that the preference for $\neg$ over * in the present paper is due solely to the fact that * has so far not found applications to natural language. From the point of view of dynamic logic, however, it is certainly interesting to consider the set Π_* of programs π given by

$$\pi ::= \varphi? \mid x :=? \mid \pi_1; \pi_2 \mid \pi_1 + \pi_2 \mid \pi^* \; .$$

Without program negation $\neg$, it becomes convenient to add more tests, and to let φ above range over a set $\Phi(L, X)$ of L-formulas (with free variables from X) that includes all atomic L-formulas, and is closed under conjunction, renaming of variables, and existential quantification. For completeness, let us state the definition

$$\alpha[\![\pi^*]\!]_M \beta \text{ iff } \alpha[\![\pi^k]\!]_M \beta \;\; \text{ for some } k \geq 0 \; ,$$

where π^0 is $\exists x \; x = x?$, and π^{k+1} is $\pi^k; \pi$.

Lemma A. *Let M be an L-model, π be a program in Π_*, and α be an $[\![M]\!]$-state with domain X_0 such that $\emptyset[\![\pi]\!]_M \alpha$. Then there is a formula $\psi_{M,\alpha,\pi} \in \Phi(L, X_0)$ satisfied by M, α such that for all L-models N and valuations $\beta : X_0 \to |N|$ satisfying $\psi_{M,\alpha,\pi}$, $\emptyset[\![\pi]\!]_N \beta$.*

Proof. Using the semantic clauses for $+$ and $\cdot^*$, extract a finite sequence $\pi_1; \pi_2; \ldots; \pi_n$ of tests and assignments that

(i) is given by ($\Rightarrow$ of) the semantic derivation of $\emptyset[\![\pi]\!]_M \alpha$, and
(ii) satisfies the following condition (by $\Leftarrow$): for every L-model N and $[\![N]\!]$-state β,

$$\emptyset[\![\pi_1; \ldots; \pi_n]\!]_N \beta \text{ implies } \emptyset[\![\pi]\!]_N \beta \; .$$

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 $\pi_1; \ldots; \pi_i$, and let $x_1^i, \ldots, x_{k_i}^i$ be fresh variables, the intuition being that x_j^i represents the value of x_j after π_i is executed. Now, appealing to the closure properties of $\Phi(L, X)$, let $\psi_{M,\alpha,\pi}$ be

$$\exists x_1^1 \cdots \exists x_{k_1}^1 \cdots \exists x_1^n \cdots \exists x_{k_n}^n \bigwedge_{1 \le j \le k_n} x_j = x_j^n \;\&\; \bigwedge_{1 < i \le n} \varphi_i \ ,$$

where for $1 < i \le n$, φ_i relates $x_1^{i-1}, \ldots, x_{k_{i-1}}^{i-1}$ to $x_1^i, \ldots, x_{k_i}^i$ after the execution of π_i. $\dashv$

Call formulas satisfying the requirements for $\psi_{M,\alpha,\pi}$ in Lemma A *(π, M)-records of α*. Given L-models M and N, write $M \equiv_{\Phi(L)} N$ when they satisfy the same sentences in $\Phi(L, \emptyset)$. Now, form the transition system $[M]_*$ from $[\![M]\!]$ as $[M]$ was, but with Π_L replaced by Π_*.

Lemma B. *Assume $M \equiv_{\Phi(L)} N$, and $\{\emptyset\}[\pi]_M s$, $\{\emptyset\}[\pi]_N s'$, where $\pi \in \Pi_*$. Then for every $\pi' \in \Pi_*$,*

$$\{\emptyset\}[\pi']_M s \text{ iff } \{\emptyset\}[\pi']_N s' \ .$$

Proof. By symmetry, it is enough to prove one direction of the bi-conditional. In fact, it suffices, again through an appeal to symmetry, to show (assuming $M \equiv_{\Phi(L)} N$, $\{\emptyset\}[\pi]_M s$ and $\{\emptyset\}[\pi']_M s$) that whenever $\emptyset[\![\pi]\!]_N \beta$, then $\emptyset[\![\pi']\!]_N \beta$.[9] So suppose $\emptyset[\![\pi]\!]_N \beta$. Call domain$(\beta)$ X_0, and let

$$\Psi(X_0, \pi') := \{\psi \in \Phi(L, X_0) \mid \exists M' \exists \alpha : X_0 \to |M'| \ \ \psi \text{ is a } (\pi', M')\text{-record of } \alpha\}$$
$$\Theta(N, \beta) := \{\theta \mid N \models \theta[\beta] \text{ and } \theta \text{ or } \neg\theta \in \Phi(L, X_0)\} \ .$$

If we can demonstrate the consistency of

$$\Theta(N, \beta) \cup \{\bigvee \Psi(X_0, \pi')\} \ ,$$

then we can conclude that $\emptyset[\![\pi']\!]_N \beta$ through an application of Lemma A. But by the General Omitting Types Theorem given in p. 108 of Keisler [15], we need only observe that for every finite piece $\Delta \subset \Theta(N, \beta)$, the conjunction

$$\psi_{N,\beta,\pi} \;\&\; \bigwedge \Delta$$

is satisfied by M relative to some $\alpha : X_0 \to |M|$ (since $M \equiv_{\Phi(L)} N$), which in turn satisfies some $\psi \in \Psi(X_0, \pi')$ (by Lemma A) since $\{\emptyset\}[\pi]_M s \ni \alpha$ and $\{\emptyset\}[\pi']_M s$. $\dashv$

Theorem C. *For L-models M and N, the following are equivalent.*

1. $M \equiv_{\Phi(L)} N$.
2. $[M]_* \cong [N]_*$.
3. $[M]_* \leftrightarrows [N]_*$.

[9] The point is that the (asymmetric) condition $\{\emptyset\}[\pi]_N s'$ is used only after establishing $\emptyset[\![\pi]\!]_N \beta$ iff $\emptyset[\![\pi']\!]_N \beta$.

Proof. $3 \Rightarrow 1$ follows from the inclusion in Π of tests for all of $\Phi(L)$. $2 \Rightarrow 3$ is trivial (since $\cong$ always implies $\leftrightarrow$). All that remains is $1 \Rightarrow 2$. Let $R_{M,N}$ be the relation

$$\{(s, s') \mid \exists \pi \ \{\emptyset\}[\pi]_M s \text{ and } \{\emptyset\}[\pi]_N s'\} \ .$$

It suffices to show that $R_{M,N} : [M]_* \cong [N]_*$, assuming $M \equiv_{\Phi(L)} N$. Proceed as follows.

1. From Lemma A conclude that $R_{M,N}$ relates $\{\emptyset\}$ to $\{\emptyset\}$. (For example, if $\emptyset[\![\pi]\!]_M \alpha$, then as $M \equiv_{\Phi(L)} N$, $N \models \exists \bar{x} \psi_{M,\alpha,\pi}$ and so for some β, $\emptyset[\![\pi]\!]_N \beta$.)
2. Moreover, by Lemma B, $R_{M,N}$ defines a function from $[M]_*$-states to $[N]_*$-states.
3. Writing f for that function, the bi-conditional

$$s_0[\pi]_M s_1 \text{ iff } f(s_0)[\pi]_N f(s_1)$$

follows from the (easily established) fact that

$$s_0[\pi; \pi']_M s_1 \text{ iff } \exists s_2 \ \ s_0[\pi]_M s_2 \text{ and } s_2[\pi']_M s_1 \ .$$

⊣

Two defects with the duality given in Theorem C are worth pointing out. First, the requirement that $\Phi(L, X)$ be closed under existential quantification cannot simply be dropped by modelling the effect of existential quantification via random assignment (i.e., defining $\exists x \varphi?$ as "$x :=?; \varphi?$") since tests have no side-effects, whereas random assignments affect the assignment of a value to a variable. Second, the Π_*-transition system $[M]_*$ is never "reduced": the $[M]_*$-state generated by $x :=?$ is different from that generated by

$$x :=? + (\exists x \ x = x \ ?) \ ,$$

even though they are related by a Π_*-bisimulation. Both defects can be corrected by closing either $\Phi(L, X)$ under modalities labelled by programs (i.e., forming formulas $[\pi]\varphi$), or (equivalently) the collection Π_* of programs under negation $\neg$. But then Lemma A breaks down (as does Theorem C) — the program

$$x :=? \ ; ([\textbf{while } x > 0 \textbf{ do } x := x - 1]0 = 1)?$$

discriminates between standard and non-standard models of arithmetic. (As suggested by the use of an omitting types argument in Lemma B, infinitary logic is not far behind.)

Thus, to stay within first-order logic, $\neg$ must be introduced only if * is dropped. Replacing Π_* by Π_L, Lemma A can be strengthened (using the finite bound on the non-determinism of programs) to

Lemma A$^\neg$. *Given a program $\pi \in \Pi_L$, and a finite subset X_0 of X, there is an L-formula $\psi_{X_0,\pi}$ with free variables in X_0 such that for every L-model N and $\beta : X_0 \to |N|$,*

$$\emptyset[\![\pi]\!]_N \beta \text{ iff } N \models \psi_{X_0,\pi}[\beta] \ .$$

Proof. Rather than reducing a program π to a sequence of primitive steps (as in the Kleene normal form theorem), a simpler reduction (in the manner of Tarski's inductive definition of satisfaction) is possible, because (in contrast to Lemma A) the bi-conditional holds. Sequential composition forces us, however, to work with an arbitrary initial state α (instead of simply $\emptyset$). Accordingly, strengthen the induction hypothesis on π as follows: for every finite subset X_0 of X and every $X' \subseteq X_0$, there is an L-formula $\chi_{X',X_0,\pi}$ with free variables from the disjoint union $X' + X_0$ of X' and X_0, and an L-formula $\theta_{X',\pi}$ with free variables from X' such that for every L-model N, $\alpha : X' \to |N|$, and $\beta : X_0 \to |N|$,

$$\alpha[\![\pi]\!]_N\beta \text{ iff } N \models \chi_{X',X_0,\pi}[\alpha + \beta]$$
$$\text{there is } no\ \gamma \text{ such that } \alpha[\![\pi]\!]_N\gamma \text{ iff } N \models \theta_{X',\pi}[\alpha] \ .$$

For the first bi-conditional, the trick is to record an un-corrupted copy of X' in $X' + X_0$ (so as to be able to store relevant conditions on the initial state α) before executing π. Note that for every π, there is a finite set X_π such that if $\alpha[\![\pi]\!]_N\gamma$ then $\text{domain}(\gamma) - \text{domain}(\alpha) \subseteq X_\pi$. This pushes through the argument not only for sequential composition, but also for the second bi-conditional. That is, $\theta_{X',\pi}$ can be constructed by taking a conjunction of formulas of the form $\neg\exists\overline{x}\ \chi_{X',X_1,\pi}$ for finitely many X_1's. $\dashv$

We might now proceed as before, proving $\neg$-versions of Lemma B and Theorem C (where $\Phi(L,X)$ is the set of all L-formulas on X). Instead, however, let us observe that under Π_L, the states generated by $x :=?$ and by $x :=? + \exists x\ x = x?$ are no longer bisimilar since $\neg(x = x?) \in \Pi_L$. In fact,

Proposition B$^\neg$. *If R is a bisimulation on $[M]$ and sRs' then $s = s'$.*

Proof. Assume R is a bisimulation on $[M]$ and sRs'. Choose π and π' that give s and s', respectively (i.e., $\{\emptyset\}[\pi]_M s$ and $\{\emptyset\}[\pi']_M s'$), and let X_1 be a finite subset of X containing all variables of X occuring in π or π'. For every subset X_0 of X_1, let π_{X_0} be the program

$$\bigwedge_{x \in X_0} x = x\ ?\ ;\ \neg(x_1 = x_1?)\ ;\ \neg(x_2 = x_2?)\ ;\ \ldots\ ;\ \neg(x_k = x_k?)\ ,$$

where $X_1 - X_0 = \{x_1, \ldots, x_k\}$, and fix L-formulas $\psi_{X_0,\pi}$ and $\psi_{X_0,\pi'}$ given by Lemma A$^\neg$, with free variables from X_0. Observe that since sRs' (where R is a bisimulation), it follows from Lemma A$^\neg$ that for every $X_0 \subseteq X_1$,

$$\neg\exists t\ \ s\ [\pi_{X_0}\ ;\ \psi_{X_0,\pi}\&\neg\psi_{X_0,\pi'}\ ?]_M\ t$$

and

$$\neg\exists t'\ \ s'\ [\pi_{X_0}\ ;\ \psi_{X_0,\pi'}\&\neg\psi_{X_0,\pi}\ ?]_M\ t' \ .$$

That is, for every $X_0 \subseteq X_1$, $\psi_{X_0,\pi'}$ and $\psi_{X_0,\pi}$ are satisfied by the same L-models M and functions from X_0 to $|M|$, whence $s = s'$ (again by Lemma A$^\neg$). $\dashv$

Hence, a bisimulation R between $[M]$ and $[N]$ must be an isomorphism (since $R \circ R^{-1}$ is a bisimulation on $[M]$), and a $\neg$-version of Theorem C can be proved by showing

$3 \Rightarrow 2 \Rightarrow 1 \Rightarrow 3$ (rather than following the direction $3 \Rightarrow 1 \Rightarrow 2 \Rightarrow 3$ in the proof above of Theorem C).

Theorem 2. *For L-models M and N, the following are equivalent.*

1. $M \equiv_L N$.
2. $[M] \cong [N]$.
3. $[M] \leftrightarrows [N]$.

Remark (for readers familiar with the model-theoretic notions involved). Prof. van Benthem has suggested replacing in Theorem 1 $M \cong N$ by M is partially isomorphic, or $L_{\infty\omega}$-equivalent to N. Under this modification, the assumption that M and N are countable can be dropped. Futhermore, the parallel with Theorem 2 can then be strengthened by bringing in the algebraic characterization of $\equiv_L$ as being so-called "finitely isomorphic" — a weakened version of partially isomorphic, also having a "back-and-forth" clause, that yields a finite variable hierarchy described, for example, in section 6 of van Benthem [4]. On the other hand, some may find the countability assumption in Theorem 1 (occurring also in Proposition 4) harmless, and a reasonable price for securing an isomorphism. Moreover, the very process of constructing (back-and-forth) an isomorphism is instructive, albeit standard.

This article was processed using the LaTeX macro package with LLNCS style