1980

Models for Verifiers

Francine Berman

Report Number:

80-343

MODELS FOR VERIFIERS

by

Francine Berman

Dept. of Computer Science
Purdue University
W. Lafayette, IN 47907

CSD-TR 343

MODELS FOR VERIFIERS

by

Francine Berman

Purdue University

The research reported here was supported in part by NSF Grant MCS77-02474. Most of the results of this paper are from the dissertation of F. Berman written at the University of Washington under the direction of R. W. Ritchie. Some of the results were presented at the 11th Annual ACM Symposium on the Theory of Computing, May 1979, under the title "A Completeness Technique for D-axiomatizable Semantics."

Abstract

We discuss three classes of models which interpret a natural proof system for Propositional Dynamic Logic. We compare their usefulness as models for verification and show that each of the three classes satisfy the same set of formulae. One of these classes can be used to give a simple proof of completeness of a natural proof system for Propositional Dynamic Logic. By the equivalence of their theories, this implies the completeness of each of the three classes of models.

Introduction

Propositional Dynamic Logic (PDL) is a formal language for reasoning about programs. As with flowchart schemes, programming constructs such as assignment are suppressed and programs in PDL are represented in a streamlined form as regular expressions with tests. Hence the language provides a description of the flow of control of a program. PDL also acts as an assertion language in which we can represent termination, partial correctness, failure conditions and loop invariance.

In this paper, we begin by describing the syntax and a proof system $\mathbb{D}$ for PDL. This proof system is natural in the sense that it characterizes the program operators U, ;, *, and ? as representations of the program operations (non-deterministic) branching, sequential execution, iteration and test. (This is analogous to the way that boolean operators - (not) and v (or) are characterized axiomatically in Propositional Calculus). We then describe three classes of interpretations (models) for PDL.

The first of the classes is called D-sound and is simply defined as the set of models in which a set of formulae D is valid. The second and third classes semantically restrict the program operators U, ; and ? to correspond to (nondeterministic) branching, sequential execution and tests.

In the class of Loop Invariant models, the program construct * represents both finite sequences of iterations and infinite halting sequences of iterations. This representation ensures the correct interpretation of loop invariant assertions.

In the class of Standard models, we restrict further and assume the construct * represents only finite sequences of iterations, i.e. we interpret * as the Kleene star (reflexive and transitive closure). For D = ⅅ, the class of ⅅ-sound models is properly contained in the the class of Loop Invariant models which is propertly contained in the class of Standard models (in order of increasing semantic restriction).

Although the D-sound models seem the most general class of interpretations, we show that for the proof system ⅅ, all three classes of models satisfy precisely the same set of formulae. Moreover, we prove a stronger result: we show that any $\mathbb{D}$-sound model can be extended to a Loop Invariant model without changing the truth-value of any PDL formula at any of the original states. This somewhat surprising result shows that the axiom schemas $\mathbb{D}$ characterize correctly the intended behavior of the programming constructs for branching, sequencing, tests and (finite and infinite) iteration.

Section 1 of this paper provides a syntactic definition of Propositional Dynamic Logic and describes the proof system $\mathbb{D}$ mentioned above. Section 2 introduces the three classes of semantics for PDL: D-sound models, Loop Invariant models and Standard models. In Section 3, we show the completeness of a proof system (with axiom schemas D) for the class of D-sound models. Section 4 concludes the technical part of the paper by showing that each $\mathbb{D}$-sound model can be extended to a Loop Invariant model and vice versa. Using the completeness result in Section 3, we obtain the completeness of the natural proof system $\mathbb{D}$ with respect to each of the classes of Loop Invariant and Standard models as a corollary. In Section 5, we draw conclusions.

Section 1: PDL and the Proof System $\mathbb{D}$

Propositional Dynamic Logic was first introduced by M. Fischer and R. Ladner in [F&L1]. The language is a simple and elegant example of the set of languages for reasoning about programs loosely based on Floyd-Hoare triples [Apt]. Many of these languages combine program descriptions with program assertions through the use of variants of the classical modal operators [] and <>. Among these languages are Pratt's Dynamic Logic ([Prattl], [Harel]), Salwicki, Mirkowska et al's Algorithmic Logic ([Sal]), Pnueli's Temporal Logic ([Pnueli]) and Manna's Modal Logic ([M & P]).

In PDL (and Propositional Algorithmic Logic), details of the program structure are suppressed and programs are represented as regular expressions with tests. The simplest formulae in PDL are propositional assertions. By using Propositional Calculus rather than Predicate Calculus for the underlying assertion language, the inherent decidability of the satisfiability of such formulae can be used to build a decision procedure to determine satisfiability of arbitrary formulae of PDL ([F&L2]). As a consequence, decision procedures for PDL may be used to partially automate verifiers in richer languages [Pratt2].

We begin by introducing the language (syntax) of PDL. We finish this section by giving a natural proof system |D for PDL.

Syntax

The basic objects of PDL are two sets of primitives:

Φ_0 the basic formulae (primitive assertions)

Σ_0 the basic programs.

Both formulae and programs in the language are constructed recursively from the basic formulae Φ_0 and the basic programs Σ_0 as follows

1) Basic programs (elements of Σ_0) are programs.

2) Basic formulae (elements of Φ_0) are formulae.

3) If p and q are formulae then -p and p v q are formulae.

4) If a and b are programs and p is a formula then a;b, aUb, a* and p? are programs.

5) If a is a program and p is a formula then <a>p and [a]p are formulae.

The notation was chosen for the natural association with operations on regular expressions, hence

aUb is intended to mean "nondeterministically execute a or b"

a;b is intended to mean "execute a then execute b" a* is intended to mean "execute a some nondeterministically chosen number of times" and

p? is intended to mean "test p and fail if false."

Expressions in PDL can be used to describe the basic statements in a simple structured programming language. In particular,

BEGIN a ; b END

can be represented as a;b,

IF p THEN a ELSE b

can be represented as p?;a U -p?;b,

WHILE p DO a

can be represented as (p?;a)*;-p?.

Formulae in PDL are intended to represent program assertions. Formulae which combine programs and assertions in one of the forms [b]p or <b>p can be used as program specifications. The modalities [b] and <b> are intended to represent the change from an initial state to a final state upon termination of a computation of program b. The box ([b]) and diamond (<b>) forms differentiate between every terminating computation of b and some terminating computation of b (the same when program b is deterministic) so that

[a]p is intended to mean "Whenever program a terminates, assertion p is true."

and

<a>p is intended to mean "Program a can terminate with assertion p true."

(Note that [] and <> are duals of each other, i.e. we intend [] to represent -<>-).

In PDL, we can express several important properties of programs.

PARTIAL CORRECTNESS

Using Floyd-Hoare triples, we can represent partial correctness and the weakest liberal precondition ([Dij]) of a program a: The triple P{a}Q corresponds to the PDL formula P→[a]Q; the weakest liberal precondition wlp(a,Q) is the PDL formula [a]Q.

TERMINATION

PDL formulae can express both the possibility and impossibility of termination, i.e.

[a]false represents "program a never terminates."

and

<a>true represents "some computation of program a terminates." (When a is deterministic, this says "a always terminates").

LOOP INVARIANCE

In PDL, we can represent both program iterations and invariant assertions. Hence loop invariance can be represented by the formula [a*]p, i.e. p is true before or after any number of iterations of program a.

A Proof System

In order to use PDL as a language for proofs of correctness, we give the proof system $\mathbb{D}$, first introduced by Segerberg [Seg1] and Parikh [Par]. This proof system is natural in the sense that the programming constructs ∪, ;, * and ? are axiomatically characterized as (nondeterministic) branching, sequential execution, (finite and infinite) iteration and tests in the same way that v and - are characterized as disjunction and negation in proof systems for Propositional Calculus.

We take as axioms for the proof system $\mathbb{D}$ all instances of the following set of schemas

$D_;$ $\langle a;b\rangle p \leftrightarrow \langle a\rangle\langle b\rangle p$

$$D_U \qquad \langle a \cup b\rangle p \leftrightarrow \langle a\rangle p \vee \langle b\rangle p$$

$$D_? \qquad \langle p?\rangle q \leftrightarrow p \wedge q$$

$$D_* \quad \left\{ \begin{array}{l} p \rightarrow \langle a^*\rangle p \\ \langle a\rangle p \rightarrow \langle a^*\rangle p \\ \langle a^*\rangle\langle a^*\rangle p \rightarrow \langle a^*\rangle p \\ \langle a^*\rangle p \rightarrow (p \vee \langle a^*\rangle(-p \wedge \langle a\rangle p)) \\ \text{(induction)} \end{array} \right.$$

$$D_0 \quad \left\{ \begin{array}{l} \langle a\rangle(p \vee q) \leftrightarrow \langle a\rangle p \vee \langle a\rangle q \\ [a](p \rightarrow q) \rightarrow ([a]p \rightarrow [a]q) \\ \text{Substitution instances of propositional tautologies.} \end{array} \right.$$

We take as rules of inference

Modus Ponens: From A and $A \rightarrow B$, infer B.

and

Necessitation: From A, infer $[a]A$.

(We will refer to both the set of formulae and the proof system described here as $\mathbb{D}$ when the context is clear).

It is interesting to note that the provable formulae of this system give a Herbrand interpretation ([Grei]) to PDL formulae with respect to the intended decomposition of the programming constructs.

Section 2: D-sound, Loop Invariant and Standard Models

In this section, we introduce three classes of models for PDL.

We first describe the class of *D-sound* models. The class of D-sound models is simply the set of models of PDL in which a given set of formulae D is valid and in which -, v, <> and [] retain their intended (usual) interpretations. This class provides a general interpretation in which the theorems represent the set of correct formulae which would be produced by a PDL verifier with the formulae D acting as program specifications. Other than preserving the correctness of these specifications, D-sound models have no other semantic restrictions on programs. (Although these constraints seem weak, we will show that for an appropriate set of formulae D (for example, $\mathbb{D}$), the validity of such formulae induce strong semantic restrictions).

The second class of models is the class of *Loop Invariant* (*LI*) models. In Loop Invariant models, programs are interpreted as regular expressions (with tests) with one important exception: the programming construct * in LI models represents both finite iteration (as with regular expressions) and infinite halting iteration. Roughly, this means that a formula [b*]p will express the loop invariance of assertion p but the program b* represents terminating computations which may include more than the set of all finite iterations of program b. In particular, the programs b* and b in Loop Invariant models are related but not in the obvious way (with * as reflexive and transitive closure). LI models have stronger semantic constraints than D-sound models but are less constrained than Standard models.

The third class is the class of Standard models. In Standard models, the programming constructs U, ;, ? and * are interpreted as operations over regular sets (where p? would be interpreted as a single symbol, the associated set representing the diagonal of pairs of states at which p was true). In particular, * operates as reflexive and transitive closure so that the program b* represents all finite iterations of program b. The class of Standard models are the most semantically constrained of the three classes but also provides the intended interpretation of programs given in Section 1 . (The class of Standard models is the class most widely associated with PDL in the literature for precisely this reason).

In spite of their apparant differences, we show in Sections 3 and 4 that all three classes are closely related.

Definition: A model M of PDL is a triple M = (W, Π, p) in which

W is a set of states,

Π is a valuation which assigns to each basic assertion a set of states (at which that assertion is true)

ρ is a valuation which assigns to each program a set of pairs (w_1,w_2) of states (where (w_1,w_2) assigned to program b means that starting in state w_1, b may terminate in state w_2).

We extend Π to interpret all formulae as follows:

$$\Pi(-p) = W - \Pi(p)$$

$$\Pi(p \lor q) = \Pi(p) \cup \Pi(q)$$

$$\Pi(\langle a\rangle p) = \{w \mid \exists v(\ (w,v) \in \rho(a) \ \land\ v \in \Pi(p)\ )\}.$$

Note that in general, a model only restricts the way in which the logical connectives and modal operators may be interpreted. An interpretation is given to every program and formula but in some models this may be completely arbitrary with respect to the way component parts of programs may interact.

Let $M = (W, \Pi, \rho)$ be a model. We let the notation $\underline{M,w \models A}$ denote the statement "w is in $\Pi(A)$".

Let M be a model. Then $\underline{Th(M)}$ is the set of PDL formulae $\{A \mid$ for all w, $M,w \models A\}$, denoted the theory of M. (If A is in Th(M), we also say $\underline{M \models A}$). Given a class of models $\overline{M}$, let $Th(\overline{M})$ denote $\bigcap_{M \in \overline{M}} Th(M)$.

Let M be a model and let w and w' be states in M. We say that w and w' are <u>indistinguishable in M</u> if for all PDL formulae p, $M,w \models p$ iff $M,w' \models p$. (When the model is clear, we simply say that w and w' are *indistinguishable*).

Definition: A *Loop Invariant* (*LI*) model is a model in which the program valuation function is constrained as follows:

$$\rho(a;b) = \rho(a) \circ \rho(b)$$

$$\rho(a \cup b) = \rho(a) \cup \rho(b)$$

$\rho(p?) = \{(w,w') \mid$ w and w' are indistinguishable and w is in $\Pi(p)\}$

$$\rho(true?) \subseteq \rho(a^*)$$

$$\rho(a) \subseteq \rho(a^*)$$

$$\rho(a^*) \circ \rho(a^*) \subseteq \rho(a^*)$$

$$\Pi(\langle a\rangle p \rightarrow (p \vee \langle a^*\rangle(-p \wedge \langle a\rangle p))) = W.$$

Loop Invariant models were called nonstandard models in [Par] and Parikh models in [Bel]. They were first introduced by R. Parikh in [Par].

Definition: A *Standard* model is a model in which the program valuation function is constrained as follows:

$$\rho(a;b) = \rho(a) \circ \rho(b)$$

$$\rho(a \cup b) = \rho(a) \cup \rho(b)$$

$\rho(p?) = \{(w,w') \mid$ w and w' are indistinguishable and w is in $\Pi(p)\}$

$\rho(a^*) = \bigcup_{n\geq 0} \rho(a^n)$ (where a^0 = true?)

Standard models were first introduced by M. Fischer and R. Ladner in [F&L1] and later in [F&L2].

Note that the class of Standard models is a proper subclass of the class of Loop Invariant models. However, these classes are closely related. By definition, star-free formulae are interpreted precisely the same way in LI and Standard models. In addition, since no single PDL formula can distinguish the representation of finite and infinite halting iteration from (unbounded) finite iteration, the classes of Loop Invariant and Standard models satisfy precisely the same theories. We note this in the following

Theorem 1

Let S be the class of Standard models and LI be the class of Loop Invariant models of PDL. Then Th(S) = Th(LI).

Proof

Let M be a Standard model. Then M is an LI model since it is straightforward to show that the induction schema

$\langle a^*\rangle p \rightarrow (p \vee \langle a^*\rangle(-p \wedge \langle a\rangle p))$

is valid in M.

That $Th(S) \subseteq Th(LI)$ was shown in [Par]. ☒

For $D = \vdash D$, it is also true that $Th(\vdash D\text{-sound}) = Th(LI)$. We demonstrate this two ways at the end of Section 4.

Section 3: Completeness

In defining the proof system $\vdash D$ in Section 1, we claimed that the axiom schemas provided a natural and useful set of program specifications with which to construct portions of an automatic verifier. In interpreting input and output specifications for such a verifier, it is important that our notion of a verification is reasonable (soundness) and that correct programs are verifiable (completeness).

The weakness of the semantic restrictions on D-sound models makes it seem plausible that a proof system based on axiom schemas D would be sound and complete with respect to the class of D-sound models. This is indeed the case as we show in this section.

We begin with some definitions. Let D be a set of formulae. Let P_D denote the proof system with axiom schemas D and rules of inference Modus Ponens and Necessitation (see Section 1). A <u>proof</u> in P_D is a sequence of formulae, each of which is an axiom or derived from previous formulae in the sequence by application of a rule of inference. The last formula in a proof is provable. Let the notation $\vdash A$ denote the statement "A is provable". A set of formulae S is inconsistent iff there is a finite subset $\{S_1, \ldots, S_n\}$ of S and a formula A with $\vdash (S_1 \wedge \ldots \wedge S_n) \rightarrow (A \wedge -A)$. A set of formulae is consistent iff it is not inconsistent. Let Pr(D) denote the set of provable formulae of P_D. (Recall that we denote the valid formulae of a class of models $\overline{M}$ by $Th(\overline{M})$).

Let D be a consistent set of formulae. Let M_D be the class of D-sound models. We will show that P_{DUD_0} is sound and complete with respect to M_D, i.e. $Pr(D \cup D_0) = Th(M_D)$. (We include the schemas D_0 as axioms in the proof system to ensure the usual behavior of -, v, <> and []. It is straightforward to show that all models are D_0-sound models). The proof of completeness uses a classical Henkin construction and is a generalization to this system of the modal techniques found in [Seg2]. As a corollary of this result and the theorems in Section 4, we will also show soundness and completeness of $\models$D with respect to the classes of LI and Standard models.

Theorem 2 (Soundness)

Let D be a consistent set of formulae. Let M_D be the class of D-sound models. Then $Pr(DUD_0) \subseteq Th(M_D)$.

Proof

It is sufficient to show that the formulae $D \cup D_0$ are valid in M_D and the rules of inference preserve validity. By definition of model, $D_0 \subseteq Th(M_D)$. By definition of D-sound, $D \subseteq Th(M_D)$. It is straightforward to show that Modus Ponens and Necessitation preserve validity. ☒

Theorem 3 (Completeness)

Let D be a consistent set of formulae. Let M_D be the class of D-sound models. Then $Th(M_D) \subseteq Pr(D \cup D_0)$.

Proof

We prove the contrapositive: If a formula A is not provable then there is some D-sound model in which -A is satisfiable (so A is not valid). In fact, we prove a slightly stronger result: There is a model N_D in which the negation of every unprovable formula is satisfiable. We construct N_D given a consistent set of formulae $D \cup D_0$ as follows:

Let $N_D = (W, \Pi, p)$ where

$W = \{w \ \{\text{all formulae}\} |$

i) For all formulae A, A is in w or -A is in w but not both.

ii) For all formulae A and B, if $A \rightarrow B$ and A are in w then B is in w.

iii) $Pr(D \cup D_0) \subseteq w$.

(i.e. W is the set of all consistent complete extensions of $D \cup D_0$).

$\Pi(p) = \{w \mid p \text{ is in } w\}$ for p in Φ_0

$\rho(a) = \{(w,v) \mid \forall A, [a]A \text{ in } w \Rightarrow A \text{ is in } v\}$

for any program a.

Extend Π to an interpretation of all formulae in the usual way (see Section 1) so that N_D is a model of PDL. Notice that N_D is essentially a modal version of the model constructed in the classical Henkin proof of the completeness of Predicate Calculus. Our proof will be analogous to this construction in that we will show that

a) N_D is a D-sound model. (N_D is in M_D).

b) The negation of every unprovable formula A is satisfiable in N_D.

To show a) and b), we first prove the following

Lemma 1

For all formulae A, and for all states w in W,

$N_D, w \models A$ iff A is in w.

Proof (After Segerberg [Seg2])

Proceed by structural induction on the formula A.

If A is a basic formula then by definition, w is in $\Pi(A)$ iff A is in w.

It is straightforward from the definitions to show that the result is true for formulae of the form $A = B \vee C$ and $A = -B$. Let $A = [a]B$.

We wish to show that $N_D, w \models [a]B$ implies that $[a]B$ is in w. Assume towards a contradiction that this is not the case. Consider the set $S = \{C \mid [a]C \text{ is in } w\}$. We wish to show that S is consistent.

Assume S is inconsistent. Then there exist formulae $C_1, \ldots, C_n$ in S and a formula E with

$\vdash C_1 \wedge \ldots \wedge C_n \rightarrow (E \wedge -E)$ by definition.

But then

$\vdash [a](C_1 \wedge \ldots \wedge C_n) \longrightarrow [a](E \wedge -E)$ and

$\vdash [a](E \wedge -E) \rightarrow [a]B$ by Necessitation and D_0.

Hence

$\vdash [a]C_1 \wedge \ldots \wedge [a]C_n \rightarrow [a]B.$

Since each state w contains all the provable formulae and by definition $[a]C_i$ is in w for each i, $[a]B$ must be in w. This contradicts our assumption that S was inconsistent.

Now consider the set $T = S \cup \{-B\}$. We would like to show that the addition of $-B$ preserves the consistency of S. The proof is analogous to the previous argument: Towards a contradiction, the only reasonable candidates for an inconsistent subset of S are sets including $-B$. We can apply the same procedure as before to derive the contradiction that $[a]B$ is in w. Hence T is consistent.

Since T is consistent with respect to $D \cup D_0$, $D \cup D_0 \cup T$ is consistent. Extend this set to a state v in W in the classical way, i.e. let $p_1, p_2, \ldots$ be an enumeration of the formulae of PDL. Let $v_0 = D \cup D_0 \cup T$. Recursively define v_{n+1} to be $v_n \cup \{p_n\}$ if this set is consistent and $v_n \cup \{-p_n\}$ otherwise. Let $v = \bigcup_{n\geq 0} v_n$. Then it is straightforward to show that v is a complete consistent set of formulae and hence a state in N_D.

Since T is a subset of v, (w,v) is in $\rho(a)$ by definition. Recall that by hypothesis we had $N_D,w\models[a]B$. Hence (w,v) in $\rho(a)$ implies that v is in $\Pi(B)$. But this provides the desired contradiction since by induction, $N_D,v\models -B$.

We have shown that $N_D,w\models[a]B$ implies that $[a]B$ is in w. For the other direction, let $[a]B$ be in w, and assume towards a contradiction the $N_D,w\models-[a]B$. Let (w,v) be an input-output pair in $\rho(a)$ and v a state at which $-B$ is true.

By definition, (w,v) in $\rho(a)$ implies that for all formulae C, [a]C is in w only if C is in v. In particular, [a]B in w implies that B is in v. Contradiction.

Hence for all formulae A, $N_D,w \models A$ iff A is in w. ⊠

We can use Lemma 1 to prove both a) and b). Since the formulae D are in every state w in N_D, $N_D,w|=D$ for all w. Hence each formula of D is valid in N_D and N_D is a D-sound model (proving a). To show b), let A be an unprovable formula. Then {-A} is consistent with respect to D U D_0. Extend D U D_0 U {-A} to a state w in N_D by the procedure given in Lemma 1. Also by Lemma 1, -A in w implies that $N_D,w \models -A$. Hence -A is satisfiable in N_D. ⊠

<u>Corollary 1</u>

$Pr(D \cup D_0) = Th(M_D)$.

A particularly nice property of N_D is that if any of the schemas in $\mathbb{D}$ are included in D, the corresponding semantic restriction on program interaction holds for the accessibility relation ρ in N_D. Recall that in the construction of N_D, program interaction was not explicitly specified by semantic constraints on ρ (i.e. for an arbitrary D-sound model M, that the schema $\langle a \cup b \rangle p \longleftrightarrow \langle a \rangle p \vee \langle b \rangle p$ is in D does not necessarily imply that $\rho(a \cup b) = \rho(a) \cup \rho(b)$ holds in M [Bc2]). We show this property for one implication of the schema $D_;$ (for sequencing) for N_D in the next proposition. (Note that in particular, if $D = |D$, $N_{|D}$ is a Loop Invariant model).

Proposition 1

Let D be a consistent set of formulae which inludes all instances of the schema

$$<a;b>p \rightarrow <a><b>p.$$

Then $p(a;b) \subseteq p(a) \circ p(b)$ in N_D.

Proof

Let (w,v) be an input-output state pair in p(a;b). Let $u' = \{<b>p| \ p \text{ in } v\} \cup \{p| \ [a]p \text{ in } w\}$.

We wish to show that u' is consistent. Towards a contradiction, the only reasonable candidate for an inconsistent subset is $\{<b>p_1, <b>p_2, \ldots, <b>p_n, q_1, \ldots, q_k\}$ where $p_1, \ldots, p_n$ are in v and $[a]q_1, \ldots, [a]q_k$ are in w. Since $<a;b>(p_1 \wedge \ldots \wedge p_n) \rightarrow <a><b>(p_1 \wedge \ldots \wedge p_n)$ is in $Pr(D \cup D_0)$ and $N_D, w \models <a;b>(p_1 \wedge \ldots \wedge p_n)$ then by Lemma 1, there is a state x with (w,x) in p(a) and $N_D, x \models <b>(p_1 \wedge \ldots \wedge p_n)$. x is a complete and consistent set which includes the formulae $<b>(p_1 \wedge \ldots \wedge p_n) \wedge q_1 \wedge \ldots \wedge q_k$ and $-(<b>p_1 \wedge \ldots \wedge <b>p_n \wedge q_1 \wedge \ldots \wedge q_k)$. Contradiction, hence u' is consistent.

Extend $u' \cup D \cup D_0$ to a complete consistent extension u. By construction, u is in N_D. We claim that (w,u) is in $\rho(a)$ and (u,v) is in $\rho(b)$. To see this, note that for all formulae p,

$[a]p$ in w implies p in u

and

p in v implies $\langle b\rangle p$ in u.

Hence (w,v) is in $\rho(a)\circ\rho(b)$. ⊠

Section 4: Semantic Constraints in $\mathcal{M}_D$

By completeness, for $D = \mathbb{D}$, the class of $\mathbb{D}$-sound models satisfies all and only the correct formulae given by the natural set of program specifications $\mathbb{D}$. The semantic constraints implied by the validity of these schemas seem weak: If we represent a model of PDL by a directed graph in which nodes represent states and edges represent programs, a $\mathbb{D}$-sound model cannot even guarantee for example that given the input-output state pair (w,v) for a branching program $a \cup b$, that (w,v) is an input-output state pair for component programs a or b. In an LI model, such a constraint is guaranteed by the semantic restriction $\rho(a\cup b) = \rho(a) \cup \rho(b)$.

It is somewhat surprising then, that every $\mathbb{D}$-sound model can be extended to a Loop Invariant model. Furthermore, we can construct such an extension so as not to change the truth value of any PDL formula at any state of the original ($\mathbb{D}$-sound) model. We exhibit this construction in Theorems 5 - 9 and prove the result in Theorem 10.

Conversely, we would like to show that every Loop Invariant model is a $\mathbb{D}$-sound model. This is trivial and demonstrates our contention that the semantic restrictions of Loop Invariant models are at least as strong as the axioms which induce them. As a corollary of these results, we will show that Th($M_{\mathbb{D}}$) = Th(LI). This, together with the general completeness proof of Section 3 will show that the proof system $\mathbb{D}$ is complete with respect to the classes of Loop Invariant and Standard models.

Theorem 4

Let M be a Loop Invariant model. Then M is a $\mathbb{D}$-sound model.

Proof

Let M be a Loop Invariant model. It is sufficient to show that each of the schema in $\mathbb{D}$ hold in M. This is straightforward and left to the reader. ⊠

We are aiming at the following result:

Theorem 10

Let M be a $\mathbb{D}$-sound model. Then there is an extension M' of M such that M' is a Loop Invariant model and for all formulae A and all states w in M,

$$M,w\models A \text{ iff } M',w\models A.$$

In fact, an even stronger version of this result is true. If we separate the axiom schemas of $\mathbb{D}$ according to their manipulation of the programming constructs U, ;, * and ?, the theorem is true for each group of schemas independently. These results are given in the following set of theorems.

Theorem 5

Let M be a $D_{;1}$-sound model where $D_{;1}$ is the schema $\langle a;b\rangle p\rightarrow\langle a\rangle\langle b\rangle p$. Then there is an extension M' of M with

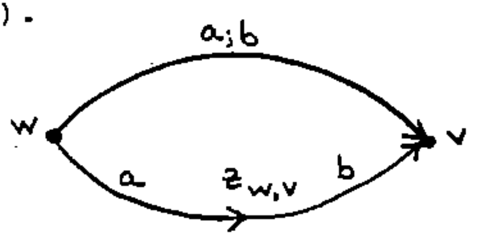
1) $\rho'(a;b)\subseteq\rho'(a)\circ\rho'(b)$

2) For all states w in M and all PDL formulae p, $M,w\models p$ iff $M',w\models p$.

Proof

Let $M = (W, \Pi, \rho)$ and let (w,v) be an input-output state pair in $\rho(a;b) - \rho(a)\circ\rho(b)$. For each such pair, we will construct a new state $z_{w,v}$ such that $(w,z_{w,v})$ is in

$\rho'(a)$ and $(z_{w,v},v)$ is in $\rho'(b)$. (We will also add other new states and edges to preserve satisfiability at w). M' will result from applying this construction to each such (w,v) in $\rho(a;b) - \rho(a)\circ\rho(b)$.



For convenience, let Th(M,x) denote $\{p|\ M,x \models p\}$ for any state x in M. Consider Th(M,w), Th(M,v) for the states w, v given above. Note that each of these sets of formulae are complete and consistent (in the sense of Section 3). In particular, we can assosciate with each state x in M, the state Th(M,x) in $N_{D;1}$. (Note that this mapping may not be one-to-one because of indistinguishable states in M. For this reason, we will graft the parts we need from $N_{D;1}$ onto M to create the extension M'). In addition, if x and y are states in M with (x,y) in $\rho(c)$ then (Th(M,x),Th(M,y)) is in $\rho_{N_{D;1}}(c)$ for any program c. Hence (Th(M,w),Th(M,v)) is in $\rho_{N_{D;1}}(a;b)$. By Proposition 1, (Th(M,w),Th(M,v)) is also in $\rho_{N_{D;1}}(a)\circ\rho_{N_{D;1}}(b)$. Hence there is a state $z_{w,v}$ in $N_{D;1}$ with $(Th(M,w),z_{w,v})$ in $\rho_{N_{D;1}}(a)$ and $(z_{w,v},Th(M,v))$ in $\rho_{N_{D;1}}(b)$. In addition, we can assume by the construction given in the proof of Proposition 1 that $\{p|\ [a]p$ is in w$\} \cup \{\langle b\rangle p|\ p$ is in v$\}$ is a subset of $z_{w,v}$.

What we've done so far is exhibited an appropriate state $z_{w,v}$ in $N_{D;1}$ so that (Th(M,w),Th(M,v)) is in

$p_{N_{D;1}}(a;b) \cap p_{N_{D;1}}(a) \circ p_{N_{D;1}}(b)$. Clearly, we want to add $z_{w,v}$ to M' for each such pair (w,v) in $p(a;b) - p(a) \circ p(b)$ so that $p'(a;b) \subseteq p'(a) \circ p'(b)$. However for each $z_{w,v}$, we need all of the formulae in the set $z_{w,v}$ to be true at $z_{w,v}$ in M'. (Note that we have denoted both states in M' and $N_{D;1}$ by the same names. To be precise, we should denote each state x in $N_{D;1}$ by another name in M' but the introduction of additional notation seems worse!) To create M', we must graft an appropriate submodel of $N_{D;1}$ onto M at each $z_{w,v}$. This is easiest to see if we consider the models M and $N_{D;1}$ as graphs.

Let G be the graph described by $N_{D;1}$, i.e the directed graph in which nodes correspond to states of $N_{D;1}$ and edges correspond to programs. Let $G_{w,v}$ be the subgraph of G in which every node lies on a path with source $z_{w,v}$. (Note that $G_{w,v} = (V_{w,v}, E_{w,v})$ where for each state x in $V_{w,v}$, there is a program c with $(z_{w,v}, x)$ in $p_{N_{D;1}}(c)$. w This will be helpful in the proof of Theorem 10).

Let $M' = (W \cup \{x \text{ in } V_{w,v} \mid (w,v) \text{ in } p(a;b) - p(a) \circ p(b)\}$, Π', $p \cup \cup\{E_{w,v} \mid (w,v) \text{ in } p(a;b) - p(a) \circ p(b)\})$. Let $\Pi'(p) = \Pi(p) \cup \cup\{x \text{ in } V_{w,v} \mid (w,v) \text{ is in } p(a;b) - p(a) \circ p(b)\}$. By Proposition 1 and construction, $p'(a;b) \subseteq p'(a) \circ p'(b)$. It is left to show that for all formulae p, $M,w \models p$ iff $M',w \models p$.

We proceed by structural induction. If p is a propositional variable then $M,w \models p$ iff $M',w \models p$ since our construction left the truth value of the propositional variables unchanged at states in M. Similarly, if p is $-q$, or $q \vee r$, it is straightforward to show that $M,w\models p$ iff $M',w\models p$ by induction.

Let $p = \langle c\rangle q$. If $c \neq a$ then $M,w\models\langle c\rangle q$ iff $M',w\models\langle c\rangle q$ by induction (and since $\rho(c) = \rho'(c)$ for $c \neq a$). Let $c = a$. Clearly, $M,w\models\langle a\rangle q$ implies $M',w\models\langle a\rangle q$. Assume $M',w\models\langle a\rangle q$. The nontrivial case is when $(w,z_{w,v})$ is in $\rho'(a)$ with $M',z_{w,v}\models q$. Assume towards a contradiction that $M,w\models-\langle a\rangle q$. Then $-q$ is in $\{q|\ M,w\models[a]q\}$. By construction, $-q$ is in $z_{w,v}$. (Note that $(z_{w,v},Th(M,v))$ is already an edge in G labelled by program b and that $z_{w,v}$ is a complete consistent set of formulae. Hence the addition of the edge $(z_{w,v},v)$ in $\rho'(b)$ in M' does not add formulae inconsistent with $z_{w,v}$). Therefore, $-q$ is in $z_{w,v}$. This provides a contradiction, since by hypothesis, $M',z_{w,v}\models q$. ☒

Theorem 6

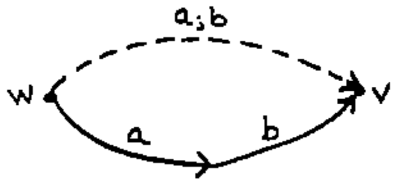
Let M be a $D_{;2}$-sound model where $D_{;2}$ is the schema $\langle a\rangle\langle b\rangle p\rightarrow\langle a;b\rangle p$. Then there is an extension M' of M with

1) $\rho'(a)\circ\rho'(b)\leq\rho'(a;b)$

2) For all states w in M and all PDL formulae p, $M,w\models p$ iff $M',w\models p$.

Proof

We construct M' from M by connecting the graph of M as follows:



Clearly $\rho'(a)\circ\rho'(b) \subseteq \rho'(a;b)$, showing 1).

For 2), we proceed by structural induction on p. If p is a propositional variable then $M,w \models p$ iff $M',w \models p$ since the construction did not alter the truth value of any propositional variable. The proof is straightforward by induction for $p = -q$ and $p = q \vee r$.

Let $p = \langle c\rangle q$. Clearly $M,w \models \langle c\rangle q$ implies $M',w \models \langle c\rangle q$ since no edges were deleted during the construction. Let (w,v) be an edge added to the graph of M. Then (w,v) is in $\rho'(a;b)$ for some programs a and b and there is a state u in M with (w,u) in $\rho(a)$ and (u,v) in $\rho(b)$. By induction, $M',v \models q$ implies $M,v \models q$. By $D_{;2}$, $M,w \models \langle a\rangle\langle b\rangle q$ implies $M,w \models \langle a;b\rangle q$. Hence $M,w \models \langle c\rangle q$. ☒

Theorem 7

Let M be a $D_{\cup}$-sound model where $D_{\cup}$ is the schema $\langle a \cup b\rangle p \leftrightarrow \langle a\rangle p \vee \langle b\rangle p$. Then there exists a model M' which extends M and in which

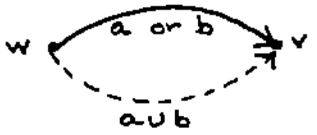
1) $p'(aUb) = p'(a)\ U\ p'(b)$

2) For all w in M and all PDL formulae p, $M,w \models p$ iff $M',w \models p$.

Proof

We construct M' from M by connecting the graph of M as follows:

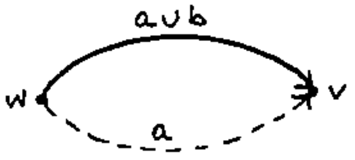
If (w,v) is in $p(a)\ U\ p(b) - p(aUb)$ then add the state pair (w,v) to $p'(aUb)$.



If (w,v) is in $p(aUb) - p(a)\ U\ p(b)$ then

add the state pair (w,v) to $p'(a)$

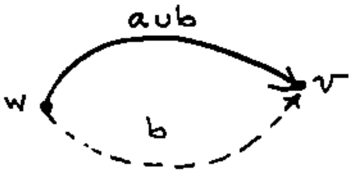
if $M,w \models [a]p$ implies $M,v \models p$ for all formulae p



and

add the state pair (w,v) to $p'(b)$

if $M,w \models [b]p$ implies $M,v \models p$ for all formulae p



We first show 2). Proceed by structural induction on p. Let p be a propositional variable. By construction, only edges were added to the graph of M. Hence the truth value of propositional variables at any state in M' remains the same. It is straightforward to show that if p is $-q$ or $q \vee r$ then 2) holds.

Let $p = \langle a\rangle q$. Since we have not deleted edges from the graph of M, clearly $M,w\models\langle a\rangle p$ implies $M',w\models\langle a\rangle p$. Conversely, let $M',w\models\langle a\rangle p$. Then there is an edge (w,v) in $\rho'(a)$ with $M',v\models q$. By induction, $M,v\models q$. If (w,v) is also in $\rho(a)$ then $M,w\models\langle a\rangle q$. If not, then (w,v) must have been added during the construction. But recall that we only added edges (w,v) for which $M,w\models[a]r$ implied $M,v\models r$ for all PDL formulae r. So in particular, $M,v\models q$ implies $M,w\models\langle a\rangle q$. Hence $M,w\models\langle a\rangle q$ iff $M',w\models\langle a\rangle q$. Similarly, if $M',w\models\langle aUb\rangle q$ then $M,w\models\langle a\rangle q \vee \langle b\rangle q$ by construction. By D_U, $M,w\models\langle a\rangle q \vee \langle b\rangle q$ implies $M,w\models\langle aUb\rangle q$.

To prove 1), we wish to show that $\rho'(aub) = \rho'(a) \cup \rho'(b)$. Clearly $\rho'(a) \cup \rho'(b) \subseteq \rho'(aUb)$. Towards a contradiction, assume that (w,v) is in $\rho'(aUb) - \rho'(a) \cup \rho'(b)$. Then it must be the case that there are formulae C and D with $M,w\models[a]C \wedge [b]D$ and $M,v\models -C \wedge -D$. But then by schemas D_U and D_0, $M,w\models[a]C \wedge [b]D$ implies $M,w\models[aUb](C \wedge D)$. This provides a contradiction since we also must have $M,v\models(C \vee D)$. Hence $\rho'(aUb) = \rho'(a) \cup \rho'(b)$. ⊠

Theorem 8

Let M be D_*-sound model where D_* is the schema $(p \vee \langle a\rangle p \vee \langle a^*\rangle\langle a^*\rangle p) \rightarrow \langle a^*\rangle p$. Then there is a model M' which extends M and in which

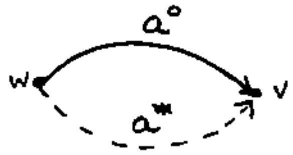
1) $\rho'(a) \cup \rho'(a^0) \cup \rho'(a^*)\circ\rho'(a^*) \subseteq \rho'(a^*)$

2) For all w in M and PDL formulae p, $M,w \models p$ iff $M',w \models p$.

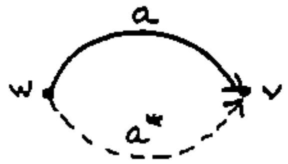
Proof

Construct M' from M by connecting the graph of M as follows:

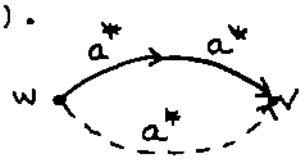
If (w,v) is in $\rho(a^0) - \rho(a^*)$ then add the state pair (w,v) to $\rho'(a^*)$.



If (w,v) is in $\rho(a) - \rho(a^*)$ then add the state pair (w,v) to $\rho'(a^*)$.



If (w,v) is in $\rho(a^*)\circ\rho(a^*) - \rho(a^*)$ then add the state pair (w,v) to $\rho'(a^*)$.



Clearly $\rho'(a) \cup \rho'(a^0) \cup \rho'(a^*) \circ \rho'(a^*) \subseteq \rho'(a^*)$. Property 2) is also straightforward: Since the construction adds only edges, $M,w \models p$ iff $M',w \models p$ for all propositional variables p. By induction, 2) holds for $p = -q$ and $p = q \vee r$.

Let $p = \langle b \rangle q$. Since we have not deleted edges from the graph, clearly $M,w \models \langle b \rangle q$ implies $M',w \models \langle b \rangle q$. Let $M',w \models \langle b \rangle q$ where (w,v) was added to $\rho'(b)$ during the construction. Then $b = a^*$ for some a and by induction, $M,v \models q$. By hypothesis, (w,v) is in $\rho(a) \cup \rho(a^0) \cup \rho(a^*) \circ \rho(a^*)$, so $M,w \models \langle a \rangle q \vee \langle a \rangle q \vee \langle a^* \rangle \langle a^* \rangle q$. By D_*, $M,w \models \langle a^* \rangle q$. Hence $M,w \models \langle b \rangle q$. ☒

Theorem 9

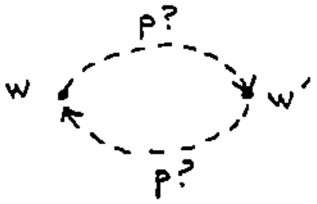
Let M be a $D_?$-sound model where $D_?$ is the schema $\langle p? \rangle q \leftrightarrow p \wedge q$. Then there is a model M' which extends M and in which

1) $\rho(p?) = \{(w,w') \mid w \text{ is in } \Pi(p) \text{ and } w \text{ and } w' \text{ are indistinguishable}\}$ for all PDL formulae p.
   (Note that if all states in M are distinguishable by PDL formulae then $\rho(p?) = \{(w,w) \mid w \text{ is in } \Pi(p)\}$).

2) For all w in M and PDL formulae q, $M,w \models q$ iff $M',w \models q$.

Proof

Let w and w' be indistinguishable states, with w in $\Pi(p)$. Then add the state pairs (w,w') and w',w) to $\rho'(p?)$.



Clearly 2) holds. For 1), let M be a $D_?$-sound model. Let (w,w') be in $\rho(p?)$. Then if w and w' are distinguishable, there exists a formula A with $M,w \models A$ and $M,w' \models -A$. Hence $M,w \models \langle p? \rangle -A$. By $D_?$, $M,w \models p \wedge -A$. Contradiction. Hence w and w' are indistinguishable. By another application of $D_?$, if (w,w') is in $\rho(p?)$ then w is in $\Pi(p)$. Hence $\rho(p?) \subseteq \{(w,w') \mid$ w is in $\Pi(p)$ and w and w' are indistinguishable$\}$. The reverse containment holds by construction. ☒

Theorems 5 - 9 can be combined in the proof of

Theorem 10

Let M be a $\mathbb{D}$-sound model. Then there is an extension M' of M such that M' is a Loop Invariant model and for all formulae p and all states w in M,

$M,w \models p$ iff $M',w \models p$.

Proof

Let M be a |D-sound model. We construct the extension model M' of M as follows:

Let M_0 = M. Repeatedly apply Theorems 5 - 9 in order generating an infinite sequence of models M_0, M_1, ... In each instance, construct model M_{i+1} from M_i by applying one of the theorems. Let M' = $\cup_{i \geq 0} M_i$, i.e. M' = (W, Π, ρ) where

$W = \cup_{i \geq 0} W_i$

$\Pi(p) = \{w \mid$ there is an i with w in $\Pi_i(p)\}$

for all formulae p

$\rho(a) = \{(w,v) \mid$ there exists an i with (w,v) in $\rho_i(a)\}$

for all programs a

We will show that M' is a Loop Invariant model and $M,w \models p$ iff $M',w \models p$ for all formulae p and all states w in M.

First note that M' is a model since Π satisfies the usual properties:

$\Pi(p \vee q) = \Pi(p) \cup \Pi(q)$

The state w is in $\Pi(p \vee q)$ iff there is an i with w in $\Pi_i(p \vee q)$, iff there is an i with w in $\Pi_i(p)$ or $\Pi_i(q)$ (since M_i is a model). Hence, w is in $\Pi(p) \cup \Pi(q)$.

The state w is in $\Pi(p) \cup \Pi(q)$ iff there is an i with w in $\Pi_i(p)$ or there is an i with w in $\Pi_i(p)$. Hence there is an i with w in $\Pi_i(p \vee q)$ and w is in $\Pi(p \vee q)$.

$\Pi(-p) = W - \Pi(p)$

The state w is in $\Pi(-p)$ iff there exists an i with w in $\Pi_i(-p)$. Assume towards a contradiction that there exists a j with w in $\Pi_j(p)$. Then let $k = \max\{i,j\}$. By construction and Theorems 5 - 9, $M_i,w \models -p$ implies $M_k,w \models -p$ and similarly, $M_j,w \models p$ implies $M_k,w \models p$. Contradiction. Hence for all i, w is not in $\Pi_i(p)$. Therefore w cannot be in $\Pi(p)$.

Assume that w is in $W - \Pi(p)$. Then for all i, w is in $\Pi_i(-p)$ since each M_i is a model. Hence w is in $\Pi(-p)$.

$\Pi(\langle a \rangle p) = \{w \mid \text{there is a } v \text{ with } (w,v) \text{ in } \rho(a) \text{ and } v \text{ in } \Pi(p)\}$

Let w be in $\Pi(\langle a \rangle p)$. Then there exists an i with w in $\Pi_i(\langle a \rangle p)$. Hence there is a v in M_i with (w,v) in $\rho_i(a)$ and v in $\Pi_i(p)$. By construction, (w,v) is in $\rho(a)$ and v is in $\Pi(p)$.

Conversely, assume that (w,v) is in $\rho(a)$ and v is in $\Pi(p)$. Then there exist i and j with (w,v) in $\rho_i(a)$ and v in $\Pi_j(p)$. Let $k = \max\{i,j\}$. Then by construction and Theorems 5 - 9, (w,v) is in $\rho_k(a)$ and v is in $\Pi_k(p)$. Hence w is in $\Pi_k(\langle a \rangle p)$ and consequently, w is in $\Pi(\langle a \rangle p)$.

We have just shown that M' is a model. To show that M' is a Loop Invariant model, we must show that the appropriate semantic constraints on p hold in M' and that the induction schema is valid in M'.

$\underline{p(a;b) = p(a) \circ p(b)}$

Let (w,v) be an input-output state pair in $p(a;b)$. Then there exists an i with (w,v) in $p_i(a;b)$. By construction, (w,v) is in $p_{i+n}(a;b)$ for all $n > 0$. Hence after the next application of Theorem 5, (w,v) is in $p_{i+n+m}(a) \circ p_{i+n+m}(b)$ for some m. Hence (w,v) is in $p(a) \circ p(b)$.

Conversely, assume that (w,u) is in $p(a)$ and (u,v) is in $p(b)$. Then there exist i and j with (w,u) in $p_i(a)$ and (u,v) in $p_j(b)$. Let $k = \max\{i,j\}$. Then by construction, (w,u) is in $p_k(a)$ and (u,v) is in $p_k(b)$. By the next application of Theorem 6, (w,v) will be in $p(a;b)$.

$\underline{p(a \cup b) = p(a) \cup p(b)}$

The proof is similar to the previous case and left to the reader.

$\underline{p(a^0) \cup p(a) \cup p(a^*) \circ p(a^*) \subseteq p(a^*)}$

Again, the proof is straightforward and left to the reader.

$\rho(p?) = \{(w,w') \mid w$ is in $\Pi(p)$ and w, w' are indistinguishable$\}$.

The proof is left to the reader.

Clearly, M' is an extension of M. Hence M' will be a Loop Invariant model once we show that the induction schema I holds in M' (i.e. $M' \models \langle a^* \rangle p \rightarrow (p \vee \langle a^* \rangle (-p \wedge \langle a \rangle p))$). First we show that for all states w in M, $M,w \models p$ iff $M',w \models p$.

Let p be a formula. Since $M = M_0$, $M,w \models p$ implies $M',w \models p$. Now suppose $M',w \models p$. Then there exists an i with $M_i,w \models p$. Assume towards a contradiction that $M_0,w \models -p$. Then by construction and Theorems 5 - 9, $M_i,w \models -p$. Contradiction since M_i is a model. Hence $M',w \models p$ and $M,w \models p$ iff $M',w \models p$.

In particular for all states w in M, $M',w \models I$ (where I is the induction schema). We would like to show that $M' \models I$. Let x be a state in M' but not in M. Then x was added during some application of Theorem 5 to a model M_i to form model M_{i+1}. Hence for some states w and v in M_i and some program c, $(z_{w,v},x)$ is in $\rho_{i+1}(c)$. In particular, (w,x) is in $\rho_{i+1}(a;c)$ for some program a. If w is in $M' - M$, we can find some descending sequence of M_i's such that eventually, there is a state u in M and programs $c, c_1, \ldots, c_k$, $a, a_1, \ldots, a_k$ such that (u,x) is in $\rho_0(a_1;c_1;\ldots;a;c)$.

Now $M_0 \models I$ so $M_0,u \models [a_1;c_1;\ldots;a;c]I$. Hence $M_{i+1},x \models I$.

Since the choice of x was arbitrary and $M',u \models [a_1;c_1;\ldots;a;c]I$, $M' \models I$. Hence M' is a Loop Invariant extension of M. ☒

Corollary 2

Let $M_{\mathbb{D}}$ be the class of $\mathbb{D}$-sound models and LI be the class of Loop Invariant models. Then $Th(M_{\mathbb{D}}) = Th(LI)$.

Proof

By Theorem 4, every $\mathbb{D}$-sound model is a Loop Invariant model. Hence $Th(LI) \subseteq Th(M_{\mathbb{D}})$.

By Theorem 10, every $\mathbb{D}$-sound model M can be embedded in a Loop Invariant model M'. Hence, $Th(M') \subseteq Th(M)$. In addition, if $M \models A$ then for all w in M, $M,w \models A$; so, for all states w in M, $M',w \models A$. We wish to show $M',w \models A$ for all w in $M' - M$. Since M is a model and A is in $Th(M)$, $[a]A$ is in $Th(M)$ for all programs a. Let w be a state in $M' - M$. By the techniques used to show $M' \models I$ in the proof of Theorem 10, there exists a program c and a state u in M with (u,w) in $p'(c)$. Since $M \models A$ implies $M,u \models [c]A$, $M',u \models [c]A$ by Theorem 10. Hence $M',w \models A$. Since w was arbitrary, $M',w \models A$ for all states in M'. Hence $M' \models A$. Therefore $Th(M) = Th(M')$ and $Th(M_{\mathbb{D}}) \subseteq Th(LI)$. ☒

Corollary 3

$P_{\mathbb{D} \cup D_0}$ is a sound and complete proof system with respect to the classes of Standard and Loop Invariant models.

Proof

By Corollary 2, $Th(M_{\mathbb{D}}) = Th(LI)$. By Theorem 1, $Th(S) = Th(LI)$. By Corollary 1, $Pr(\mathbb{D} \cup D_0) = Th(M_{\mathbb{D}})$. ⊠

Section 5: Conclusion

The main focus of this paper has been to develop alternative classes of models for PDL and their relationship to the class of Standard models for PDL. Parikh first described a nonstandard class of models for PDL in order to prove completeness for $\mathbb{D}$ with respect to the Standard models in [Par]. Most recently, Pratt discusses the use of nonstandard models for verification in Dynamic Logic [Pratt2]. But to our knowledge, this work is the first to focus on the semantic constraints induced by the regular program operators U, ;, * and ? in PDL and to focus on the relationship between the class of Standard models and alternative classes of models for PDL. We also present a new class of interpretations for PDL, the class of D-sound models.

The results presented here show a flexibility in interpretations for PDL. The class of Standard models reflects most strongly our intuitive perception of how to represent flowchart schemes but the representation of unbounded finite iteration (the construct represented by *) makes decision procedures for the language unwieldy ([F&L2]) and completeness results difficult to prove ([Par], [Pratt3]). The class of Loop Invariant interpretations would probably simplify decision procedures and satisfies the same set of true formulae as the class of Standard models. The natural proof system $\mathbb{D}$ can also be shown to be complete with respect to the class of Loop Invariant models without the aid of a small model theorem [Par].

The class of D-sound models presented here yields a general completeness technique for any set of consistent formulae which would characterize the behavior of programming constructs (as the schemas $\mathbb{D}$ characterize the intended behavior of U, ;, * and ?). This is applied to the schemas $\mathbb{D}$ to derive completeness of the class of $\mathbb{D}$-sound models in Section 3 and of the classes of Standard and Loop Invariant models in Section 4. Although completeness of the classes of Standard and Loop Invariant models is not new ([Par], [K&P], [Pratt3], etc.), our method of proof can be easily expanded to include new programming constructs such as // (shuffle) for parallel programs, -1 (reverse), $\cap$ (intersection), etc. In addition, the construction in the proofs of Theorems 5 - 9 sheds light on the nature of the constraints induced by schemas $\mathbb{D}$ on the graphs of models of PDL.

Theorems 5 - 10 show the close relationship between $\mathbb{D}$-sound and Loop Invariant models. Given that semantic constraints on state-to-state transitions cannot be completely specified in the language of PDL, Theorem 10 shows that we can nonetheless come quite close to describing these transitions. In addition, when such models fail to satisfy these constraints, they do so by leaving out state-to-state transitions rather than by including inappropriate ones. Such results also support the naturalness of the set of axiom schemas $\mathbb{D}$.

The development and study of nonstandard classes of models is motivated by their natural interpretation of PDL formulae and by their simple description and ease of technical manipulation. We hope to have contributed to that study here.

Acknowledgments

We would like to thank Mike Fischer and Bob Ritchie for their generous support and encouragement during this work. We would also like to thank Mike O'Donnell and Mitch Wand for their helpful comments on a preliminary draft of this paper.

We are grateful to Joe Halpern and Albert Meyer for pointing out an error in the original proof of Theorem 5. The present proof is based on a suggestion of Rohit Parikh.

Bibliography

[Apt] Apt, K. R., "Ten Years of Hoare's Logic, A Survey." Manuscript, Erasmus University, Rotterdam, The Netherlands, 1979.

[Be1] Berman, F., "A Completeness Technique for D-Axiomatizable Semantics." 11th Symposium on the Theory of Computing, 1979.

[Be2] Berman, F., "Syntactic and Semantic Structure in Propositional Dynamic Logic." Ph.D. Dissertation, University of Washington, 1979.

[Dij] Dijkstra, E. W., *A Discipline of Programming*. Prentice-Hall, New York, 1976.

[F&L1] Fischer, M. J. and R. E. Ladner, "Propositional Modal Logic of Programs". 9th Symposium on the Theory of Computing, 1977.

[F&L2] Fischer, M. J. and R. E. Ladner, "Propositional Dynamic Logic of Regular Programs." JCSS 18:2, April, 1979.

[Grei] Greibach, S., *Theory of Program Structures: Schemes, Semantics and Verification*. Lecture Notes in Computer Science No. 36, Springer-Verlag, New York, 1975.

[Harel] Harel, D., *First-Order Dynamic Logic*. Lecture Notes in Computer Science No. 68, Springer-Verlag, New York/Berlin, 1979.

[K&P] Kozen, D. and R. Parikh, "An Elementary Proof of the Completeness of PDL." TCS, to appear (also IBM Report RC8097, Jan. 1980).

[M&P] Manna, Z. and A. Pnueli, "The Modal Logic of Programs." 6th International Colloquium on Automata, Languages and Programming, Graz, Austria, 1979.

[Par] Parikh, R., "A Completeness Result for Propositional Dynamic Logic." Symposium on the Mathematical Foundations of Computer Science, Zakopane, Poland, 1978.

[Pnueli] Pnueli, A., "The Temporal Logic of Programs." 19th IEEE Symposium on the Foundations of Computer Science, 1977.

[Prattl] Pratt, V., "Semantical Considerations on Floyd-Hoare Logic." 17th IEEE Symposium on the Foundations of Computer Science, 1976.

[Pratt2] Pratt, V., "On Specifying Verifiers." 7th Symposium on the Priniciples of Programming Languages, 1980.

[Pratt3] Pratt, V., "A Practical Decision Method for Propositional Dynamic Logic." 10th Symposium on the Theory of Computing, 1978.

[Sal] Salwicki, A., "On Algorithmic Logic and its Applications." Technical Report, Polish Academy of Sciences, Warsaw, Poland.

[Segl] Segerberg, K., "A Completeness Theorem in the Modal Logic of Programs." Preliminary Report, Notices of the AMS, 24, 6, A-552, Oct., 1977.

[Seg2] Segerberg, K., "An Essay in Classical Modal Logic." (Volume 1), Ph.D. Dissertation, Uppsala Universitet, 1971.