The Deducibility Problem in Propositional Dynamic Logic*

Albert R. Meyer and Robert S. Streett — Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, Massachusetts USA

Grazyna Mirkowska — Institute of Mathematics, Warsaw University, Warsaw, Poland

Abstract: The problem of whether an arbitrary formula of Propositional Dynamic Logic (*PDL*) is deducible from a fixed axiom scheme of *PDL* is Π_1^1-complete. This contrasts with the decidability of the problem when the axiom scheme is replaced by any single *PDL* formula.

1 Introduction

Propositional Dynamic Logic (*PDL*) [1] is an extension of propositional logic in which "before-after" assertions about the behavior of regular program schemes can be made directly. Propositional calculus and versions of propositional modal logic, propositional temporal logic, and Propositional Algorithmic Logic [2] are all embeddable in *PDL*, but *PDL* nevertheless has a validity problem decidable in (deterministic) exponential time [4].

In this paper we consider the *deducibility problem* for *PDL*, namely the problem of when a formula p follows from a set Γ of formulae. The problem comes in two versions:

(1) p is *implied* by Γ if and only if $\wedge\Gamma \rightarrow p$ is valid.

(2) p can be *inferred* from Γ if and only if p is valid in all structures for which $\wedge\Gamma$ is valid.

Note that if p is implied by Γ then it can be inferred from Γ, but the converse does not hold in general.

For a finite set Γ, the question whether p is implied or inferred from Γ reduces to whether a formula of *PDL* is valid and so is decidable. However, axiomatizations of logical languages such as the propositional calculus or *PDL* are often given in terms of *axiom schemes*, namely, formulae whose variables may be replaced by arbitrary formulae. Thus, a single axiom scheme actually represents the *infinite* set of all formulae which are substitution instances of the scheme. Our main result is that

> *the problem of whether an arbitrary PDL formula p is deducible from a single fixed axiom scheme is of extremely high degree of undecidability, namely* Π_1^1*-complete.*

This result appears unexpected for at least two reasons. First, the easily recognizable infinite set of substitution instances of a single scheme seems initially to provide little more power than a single formula. For example, the problem of whether a single *PDL* scheme is a sound axiom, i.e., whether all its substitution instances are valid, is equivalent to the question of whether the scheme itself regarded as a formula is valid. Hence it is decidable whether a scheme is sound.

Second, many familiar logical languages satisfy the *compactness property*, namely, that if p is deducible from Γ, then in fact p is deducible from a finite subset of Γ. It follows directly from compactness that the deducibility problem from Γ is recursively enumerable relative to Γ and the set of valid formulae of the language. Since the set Γ obtained from a single axiom scheme and the set of valid formulae of *PDL* are each decidable, compactness of *PDL* would imply that the deducibility problem was recursively enumerable, whereas Π_1^1-completeness in fact implies that the deducibility problem for *PDL* is not even in the arithmetic hierarchy. This provides a dramatic illustration of the familiar fact that *PDL* is not compact.

The idea of our proof is based on an observation of Salwicki and Pratt [2] that with a finite set of axiom schemes one can essentially define the integers up to isomorphism. This idea is extended below to define structures isomorphic to the five dimensional nonnegative integer grid with coordinatewise successor and predecessor functions and an arbitrary monadic predicate. Program schemes interpreted over these grids can compute arbitrary recursive functions of integer and monadic predicate variables. The validity of formulae asserting termination of program schemes corresponds to the validity of arithmetic formulae asserting the existence of roots of such recursive functions. Validity of such arithmetic formulae with predicate variables is well known to be a Π_1^1-complete problem.

In the next section we review the syntax and semantics of *PDL* and give formal definitions of the implication and inference problems from axiom schemes. In Section 3 we define the structures called *grids* and show that they are precisely characterized by a single axiom scheme. This easily yields the main result in Section 4 that the deducibility problems are Π_1^1-complete for *PDL* schemes. The argument is then sharpened to show that Π_1^1-completeness of the inference problem holds even for restricted versions of *PDL*, namely, *test-free PDL* and *deterministic PDL* with *atomic tests*. Similarly, the implication problem is Π_1^1-complete for *test-free PDL*. Section 5 lists some open problems and related results.

2 Propositional Dynamic Logic

We are given a set of atomic programs Π_0 and a set of atomic propositions Φ_0. Capital letters $A, B, C, \ldots$ from the beginning of the alphabet will be used to denote elements of Π_0, and capital letters $P, Q, R, \ldots$ from the middle of the alphabet will be used to denote elements of Φ_0.

Definition: The set of programs, Π, and the set of formulae, Φ, of *propositional dynamic logic* (*PDL*) are defined inductively as follows (note the use of letters $a, b, c, \ldots$ to denote elements of Π and $p, q, r, \ldots$ to denote elements of Φ):

Π:
(1) $\Pi_0 \subseteq \Pi$ and $\theta \in \Pi$
(2) If $a, b \in \Pi$ then $a;b,\ a\cup b,\ a^* \in \Pi$
(3) If $p \in \Phi$ then $p? \in \Pi$

Φ:
(1) $\Phi_0 \subseteq \Phi$
(2) If $p, q \in \Phi$ then $\neg p,\ p\wedge q \in \Phi$
(3) If $a \in \Pi$ and $p \in \Phi$ then $\langle a\rangle p \in \Phi$

Definition: A *PDL structure* is a triple $S = \langle U, \models_S, \langle\rangle_S\rangle$ where

(1) U is a non-empty set, the universe of states.

(2) $\models_S$ is a satisfiability relation on the atomic propositions, i.e. a predicate on $U \times \Pi_0$.

(3) $\langle\rangle_S$ maps each atomic program A to a binary relation $\langle A\rangle_S$ on states, i.e., $\langle A\rangle_S \subset U \times U$.

Definition: For any structure S, the relation $\models_S$ and map $\langle\rangle_S$ can be extended to arbitrary formulae and programs as follows:

(1) $u \models_S \neg p$ iff not $u \models_S p$.
(2) $u \models_S p \wedge q$ iff $u \models_S p$ and $u \models_S q$.
(3) $u \models_S \langle a\rangle p$ iff $\exists v.\ u\langle a\rangle_S v$ & $v \models_S p$.
(4) $u\langle \theta\rangle_S v$ for no u, v.
(5) $u\langle a;b\rangle_S v$ iff $\exists w.\ u\langle a\rangle_S w$ and $w\langle b\rangle_S v$.
(6) $u\langle a \cup b\rangle_S v$ iff $u\langle a\rangle_S v$ or $u\langle b\rangle_S v$.
(7) $u\langle a^*\rangle_S v$ iff $u\langle a\rangle_S{}^* v$, where $\langle a\rangle_S{}^*$ is the reflexive transitive closure of $\langle a\rangle_S$.
(8) $u\langle p?\rangle_S v$ iff $u = v$ and $u \models_S p$.

The standard semantics for *PDL* given above fix the meaning of the program θ as the empty program. If a and b are two programs, then $a;b$ is the program in which a is followed by b. The program $a \cup b$ permits the nondeterministic choice of either a or b. The program a^* permits a nondeterministic choice of some number (possibly zero) of repetitions of a. If p is a formula, then $p?$ is a test or guard program which acts as the identity program if p is true and acts as the empty program θ otherwise.

Notation: If Γ is a set of formulae, then we write $u \models_S \Gamma$ if and only if $u \models_S p$ for every $p \in \Gamma$.

Definition: If p is a formula and $S = \langle U, \models_S, \langle\rangle_S\rangle$ is a structure, then p is *valid* in S if and only if $u \models_S p$ for all $u \in U$. If Γ is a set of formulae, then Γ is *valid* in S if and only if every formula in Γ is valid in S. We say that Γ *implies* p if and only if for all structures S and states u, if $u \models_S \Gamma$ then $u \models_S p$. We say that Γ *infers* q if and only if q is valid in every structure in which Γ is valid.

Remark: If Γ implies p then Γ infers p, but the converse does not hold in general.

Definition: If p and q are formulae and Q is a primitive proposition, then $p_Q{}^q$ is the formula obtained by substituting q simultaneously for every occurrence of Q in p. If L is a set of formulae, then $p_Q{}^L$ is the set of formulae obtainable by substituting an arbitrary formula of L for Q in p, i.e., $p_Q{}^L = \{p_Q{}^q \mid q \in L\}$.

Definition: The *scheme implication problem* for a set of formulae L is to determine, for given formulae p and q and primitive proposition Q, whether $p_Q{}^L$ implies q. The *scheme inference problem* for L is to determine whether $p_Q{}^L$ infers q.

It is technically convenient, given a structure, to identify or *collapse* states which are indistinguishable by formulae.

Definition: If $S = \langle U, \models_S, \langle\rangle_S\rangle$ is a structure and L is a set of formulae, then the *L-collapse of S* is the structure $T = \langle V, \models_T, \langle\rangle_T\rangle$, where the elements of V are equivalence classes of U modulo L, where u is equivalent to v modulo L if and only if u and v satisfy exactly the same formulae of L. For atomic propositions P and equivalence classes $[u] \in V$, we define the satisfaction relation $\models_T$ by the condition $[u] \models_T P$ iff $\exists v \in [u].\ v \models_S P$. For atomic programs A and equivalence classes $[u], [v] \in V$, we define the map $\langle\rangle_T$ by the condition $[u]\langle A\rangle_T[v]$ iff $\exists w \in [u].\ \exists z \in [v].\ w\langle A\rangle_S z$.

Lemma 2.1: If T is the *PDL*-collapse of a structure S, then for all *PDL* formulae p and states u of S, $u \models_S p$ iff $[u] \models_T p$.

Proof: Straightforward, by structural induction on formulae. ∎

It will be convenient to consider structures in which there is a designated initial state u, and the entire universe is accessible from u by programs using a given set of primitives.

Definition: If $S = \langle U, \models_S, \langle\rangle_S\rangle$, $u_0 \in U$, and α is a set of atomic programs, then the α-*cut* of S from u_0 is the structure $T = \langle V, \models_T, \langle\rangle_T\rangle$, where $V = \{u \in U \mid u_0\langle(A_1 \cup \cdots \cup A_n)^*\rangle_S u$ for some $A_1, \ldots, A_n \in \alpha\}$. We let $u \models_T P$ iff $u \models_S P$ and we let $u\langle A\rangle_T v$ iff $A \in \alpha$ and $u\langle A\rangle_S v$.

Lemma 2.2: Suppose that T is the α-cut from the state u of some structure S and that α contains all the atomic programs appearing in some *PDL* formula p. Then for all states v of T, $v \models_T p$ if and only if $v \models_S p$.

Proof: Straightforward, by structural induction on formulae. ∎

Corollary 2.3: If α contains all the atomic programs appearing in a *PDL* formula p, then for all structures S, p is valid in S if and only if p is valid in all the α-cuts of S.

Proof: Follows immediately from *Lemma 2.2*. ∎

3 Characterizing the Integer Grid by an Axiom Scheme

Notation: We define the following familiar and convenient abbreviations:

$$[a]q =_{\mathrm{df}} \neg\langle a\rangle\neg q$$
$$\lambda =_{\mathrm{df}} \theta^*$$
$$p \vee q =_{\mathrm{df}} (\neg p)\wedge(\neg q)$$
$$p \rightarrow q =_{\mathrm{df}} (\neg p)\vee q$$
$$p \leftrightarrow q =_{\mathrm{df}} (p\rightarrow q)\wedge(q\rightarrow p)$$
$$\mathit{true} =_{\mathrm{df}} P\rightarrow P$$
$$\mathit{false} =_{\mathrm{df}} \neg\mathit{true}$$
$$a^0 =_{\mathrm{df}} \lambda$$
$$a^n =_{\mathrm{df}} a; \cdots ;a \ (n\ a\text{'s, for } n > 0)$$
$$\mathit{if}\ p\ \mathit{then}\ a\ \mathit{else}\ b =_{\mathrm{df}} (p?;a)\cup(\neg p?;b)$$
$$\mathit{while}\ p\ \mathit{do}\ a =_{\mathrm{df}} (p?;a)^*;\neg p?$$

For the remainder of this paper let $\alpha = \{A_1, A_2, A_3, A_4, A_5, B_1, B_2, B_3, B_4, B_5\}$ be a fixed set of atomic programs and let Q and R be fixed atomic propositions. For $1 \leq i \leq 5$, let $zero_i$ be an abbreviation for $[B_i]\mathit{false}$ and let $zero$ be an abbreviation for $\bigwedge_{1\leq i\leq 5} zero_i$.

Notation: N^5 is the set of quintuples of natural numbers. We will use variables $x, y, \ldots$ to denote vectors $\langle x_1, x_2, x_3, x_4, x_5\rangle, \langle y_1, y_2, y_3, y_4, y_5\rangle, \cdots$. The five successor functions $\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5$ are defined by $y = \sigma_i(x)$ if and only if $y_i = x_i+1$ and $y_j = x_j$ for $j \neq i$.

A *canonical grid* is a structure $S = \langle N^5, \models_S, \langle\rangle_S\rangle$ such that A_i acts like σ_i, B_i acts like the inverse of σ_i (so that $zero_i =_{df} [B_i]false$ is true only at vectors whose i^{th} coordinate is zero), and R depends only on the first coordinate of vectors. A *grid* is any structure isomorphic to a canonical grid; we give a formal definition below.

Definition: A *grid* is a structure $S = \langle U, \models_S, \langle\rangle_S\rangle$ with a bijection φ: $U \rightarrow N^5$ such that:

(1) For all $u, v \in U$, $u\langle A_i\rangle_S v$ if and only if $\varphi(v) = \sigma_i(\varphi(u))$.

(2) For all $u, v \in U$, $u\langle B_i\rangle_S v$ if and only if $\varphi(u) = \sigma_i(\varphi(v))$.

(3) For all $u \in U$, if $u \models_S R$ then $v \models_S R$ for all v such that $\varphi(v)_1 = \varphi(u)_1$.

Definition: Let *grid-scheme* be an abbreviation for the conjunction of the following formulae:

zero-axiom: $\langle B_1{}^*;B_2{}^*;B_3{}^*;B_4{}^*;B_5{}^*\rangle zero$
identity-axiom: $\bigwedge_{1\leq i\leq 5} \langle A_i\rangle\langle B_i\rangle true$
AB-axiom: $\bigwedge_{1\leq i\neq j\leq 5} (\langle A_i\rangle\langle B_j\rangle true \leftrightarrow \langle B_j\rangle\langle A_i\rangle true)$
BB-axiom: $\bigwedge_{1\leq i,j\leq 5} (\langle B_i\rangle\langle B_j\rangle true \leftrightarrow \langle B_j\rangle\langle B_i\rangle true)$
R-axiom: $R \rightarrow \bigwedge_{2\leq i\leq 5} ([A_i]R \wedge [B_i]R))$

determinism-scheme: $\bigwedge_{1\leq i\leq 5} (\langle A_i\rangle Q \rightarrow [A_i]Q)$
identity-scheme: $\bigwedge_{1\leq i\leq 5} (Q \rightarrow [A_i;B_i]Q)$
AA-scheme: $\bigwedge_{1\leq i,j\leq 5} (\langle A_i;A_j\rangle Q \rightarrow [A_j;A_i]Q)$
AB-scheme: $\bigwedge_{1\leq i\neq j\leq 5} (\langle A_i;B_j\rangle Q \rightarrow [B_j;A_i]Q)$
BB-scheme: $\bigwedge_{1\leq i,j\leq 5} (\langle B_i;B_j\rangle Q \rightarrow [B_j;B_i]Q)$

The proof of the following proposition is straightforward, but quite lengthy; the reader may wish to proceed directly to section 4.

Proposition 3.1: The grids are precisely (up to isomorphism) the α-cuts of *PDL*-collapses of structures S in which $grid\text{-}scheme_Q{}^{PDL}$ is valid.

Proof: It is straightforward to verify that $grid\text{-}scheme_Q{}^{PDL}$ is valid in every grid and that every grid is (isomorphic to) the α-cut of the *PDL*-collapse of a grid.

For the converse, suppose that $T = \langle V, \models_T, \langle\rangle_T\rangle$ is the α-cut from an equivalence class $[u_{start}]$ of the *PDL*-collapse of a structure $S = \langle U, \models_S, \langle\rangle_S\rangle$ in which $grid\text{-}scheme_Q{}^{PDL}$ is valid. We shall show that T is a grid. *Lemmas 3.2* through *3.13* will establish the existence of a bijection φ: $V \rightarrow N^5$ which makes T a grid.

Lemma 3.2: There is an equivalence class $[u_{zero}] \in V$ such that $[u_{zero}] \models_T zero$.

Proof: Since $grid\text{-}scheme_Q{}^{PDL}$ is valid in S, *zero-axiom* is valid in S, hence $u_{start} \models_S \langle B_1{}^*;B_2{}^*;B_3{}^*;B_4{}^*;B_5{}^*\rangle zero$. Hence there is a state $u_{zero} \in U$ such that $u_{start}\langle B_1{}^*;B_2{}^*;B_3{}^*;B_4{}^*;B_5{}^*\rangle_S u_{zero}$ and $u_{zero} \models_S zero$. Then $[u_{zero}] \models_T$

zero, since T is the α-cut from $[u_{start}]$ of the *PDL*-collapse of S. ∎

Definition: An *AB-program* is any program of the form $a_1; \ldots ; a_n$, where each a_j is λ or an A_i or a B_i. An *A-program* is simply an *AB*-prorgram without any B_i's. A *canonical A-program* is an *A*-program of the form $A_1^{x_1};A_2^{x_2};A_3^{x_3};A_4^{x_4};A_5^{x_5}$ for some $x_1, x_2, x_3, x_4, x_5 \geq 0$. We abbreviate $A_1^{x_1};A_2^{x_2};A_3^{x_3};A_4^{x_4};A_5^{x_5}$ by *prog*(x).

Lemma 3.3: If $[u] \in V$ and a is an *A*-program, then there is at least one $[v]$ such that $[u]\langle a\rangle_T[v]$.

Proof: We first prove this lemma for the case where a is A_i for some i. By *identity-axiom*, $u \models_S \langle A_i\rangle\langle B_i\rangle true$, so that there is at least one $v \in U$ such that $u\langle A_i\rangle_S v$. Then $[u]\langle A_i\rangle_T[v]$, since T is an α-cut of the *PDL*-collapse of S. The lemma can now be proved for arbitrary *A*-programs by an easy induction on the length of programs. ∎

Lemma 3.4: If $[u] \in V$ and a is an *A*-program, then there is at most one $[v]$ such that $[u]\langle a\rangle_T[v]$.

Proof: We first prove this lemma for the case where a is A_i for some i. Suppose that $[u]\langle A_i\rangle_T[v]$ and $[u]\langle A_i\rangle_T[w]$. Then $u\langle A_i\rangle_S v$ and $u\langle A_i\rangle_S w$. Let q be any formula such that $v \models_S q$, so that $u \models_S \langle A_i\rangle q$. By *determinism-scheme*, $u \models_S \langle A_i\rangle q \rightarrow [A_i]q$. Since $u \models_S \langle A_i\rangle q$, $u \models_S [A_i]q$, so $w \models_S q$. Hence v and w agree, in S, on all formulae, so $[v] = [w]$. Therefore there is at most one $[v]$ such that $[u]\langle A_i\rangle_T[v]$. The lemma can now be proved for arbitrary *A*-programs by an easy induction on the length of programs. ∎

Lemma 3.5: If a is an *A*-program and b is any program and $[u]\langle a\rangle_T[v]$ and $[u]\langle a;b\rangle_T[w]$, then $[v]\langle b\rangle_T[w]$.

Proof: If $[u]\langle a;b\rangle_T[w]$ then there is a $[z]$ such that $[u]\langle a\rangle_T[z]$ and $[z]\langle b\rangle_T[w]$. By *Lemma 3.4*, it follows from $[u]\langle a\rangle_T[v]$ and $[u]\langle a\rangle_T[z]$ that $[v] = [z]$. So $[v]\langle b\rangle_T[w]$. ∎

Definition: Given two programs a and b, we say that a and b are *T*-equivalent if and only if $\langle a\rangle_T = \langle b\rangle_T$, i.e., for all states u and v, $u\langle a\rangle_T v$ iff $u\langle b\rangle_T v$.

Lemma 3.6: The program $A_i;B_i$ is *T*-equivalent to the identity program λ.

Proof: By *identity-axiom*, $u \models_S \langle A_i\rangle\langle B_i\rangle true$. Hence there is a state $w \in U$ such that $u\langle A_i\rangle_S w$ and $w \models_S \langle B_i\rangle true$. Hence there is a v such that $w\langle B_i\rangle_S v$ and $u\langle A_i;B_i\rangle_S v$. Now let v be any state in U such that $u\langle A_i;B_i\rangle_S v$. Let q be any formula such that $u \models_S q$. By *identity-scheme*, $u \models_S q \rightarrow [A_i;B_i]q$. Since $u \models_S q$, $u \models_S [A_i;B_i]q$, so $v \models_S q$. Hence u and v agree, in S, on all formulae, so $[u] = [v]$. Therefore, $A_i;B_i$ is the identity program in the *PDL*-collapse of S, hence also in T. ∎

Lemma 3.7: If a and b are *A*-programs and a is a permutation of b, then a and b are *T*-equivalent.

Proof: By an induction on the length of a and b, using *AA-scheme*. ∎

Lemma 3.8: If a is an *AB*-program not containing A_i, then $a;B_i$ and $B_i;a$ are *T*-equivalent.

Proof: By an induction on the length of a, using *AB-axiom*, *BB-axiom*, *AB-scheme*, and *BB-scheme*. ∎

Lemma 3.9: If a is an *AB* program not containing A_1 or B_1 and if $[u]\langle a\rangle_T[v]$, then $[u] \models_T R$ if and only if $[v] \models_T R$.

Proof: By an induction on the length of a, using *R-axiom*. ∎

Definition: An AB program a is *nonnegative* if and only if every prefix of a contains at least as many A_i's as B_i's, for $1 \leq i \leq 5$.

Lemma 3.10: Every nonnegative AB-program is T-equivalent to an A-program. ∎

Proof: If a is a nonnegative AB-program, then a is T-equivalent to $b;A_i;c;B_i;d$ where b and c are (possibly trivial) A-programs, c contains no A_i's, and d is an AB-program. By *Lemma 3.8*, a is T-equivalent to $b;A_i;B_i;c;d$, and by *Lemma 3.6*, a is T-equivalent to $b;c;d$, which is nonnegative and contains one less B_i than a. The lemma follows by an easy induction on the number of B_j's in a. ∎

Lemma 3.11: If the AB-program a is not nonnegative, then there is no $[u]$ such that $[u_{zero}]\langle a\rangle_T[u]$.

Proof: If a is not nonnegative, then a is equivalent to $b;B_i;c$ where b and c are AB-programs such that b contains no A_i's. By *Lemma 3.8*, a is T-equivalent to $B_i;b;c$. Since $u_{zero} \models_S zero$, there can be no u such that $u_{zero}\langle B_i\rangle_S u$, hence no u such that $u_{zero}\langle a\rangle_S u$, since a is T-equivalent to $B_i;b;c$. Hence there is no $[u]$ such that $[u_{zero}]\langle a\rangle_T[u]$. ∎

For the rest of the proof of *Proposition 3.1*, we will use $u, v, w, \ldots$ to denote elements of V, since we no longer need to make use of the fact that elements of V are equivalence classes of elements of U. Let u_{zero} be that element of V such that $u_{zero} \models_T zero$.

Lemma 3.12: For all $u \in V$, there is at most one x such that $u_{zero}\langle prog(x)\rangle_T u$.

Proof: Suppose $x \neq y$, but $u_{zero}\langle prog(x)\rangle_T u$ and $u_{zero}\langle prog(y)\rangle_T u$. Without loss of generality we can suppose that $x_1 > y_1$. $prog(y);B_1^{x_1}$ is not nonnegative, so by *Lemma 3.11*, there is no v such that $u_{zero}\langle prog(y);B_1^{x_1}\rangle_T v$, hence no v such that $u\langle B_1^{x_1}\rangle_T v$. Therefore $u \models_T [B_1^{x_1}]false$. $prog(x);B_1^{x_1}$ is, by *Lemmas 3.8* and *3.6*, T-equivalent to $prog(z)$ for some z. By *Lemma 3.3*, there is a w such that $u_{zero}\langle prog(z)\rangle_T w$ and hence such that $u_{zero}\langle prog(x);B_1^{x_1}\rangle_T w$. By *Lemma 3.5*, $u\langle B_1^{x_1}\rangle_T w$. Hence $u \models_T \langle B_1^{x_1}\rangle true$, a contradiction. So $x \neq y$ is not possible. ∎

We now prove that the relation between a state $u \in V$ and a vector x defined by $u_{zero}\langle prog(x)\rangle_T u$ is the desired bijection.

Lemma 3.13: There is a bijection $\varphi\colon V \rightarrow N^5$ such that $\varphi(u) = x$ if and only if $u_{zero}\langle prog(x)\rangle_T u$.

Proof: Let $u \in V$. Since T is an α-cut, there is an AB-program a such that $u_{zero}\langle a\rangle_T u$. By *Lemma 3.11*, a must be nonnegative. By *Lemma 3.10*, a is T-equivalent to some A-program b, which, by *Lemma 3.7*, is T-equivalent to $prog(x)$ for some x. By *Lemma 3.12*, x is unique, so we may define $\varphi(u) = x$. To show that φ is an injection, suppose that $\varphi(u) = \varphi(v) = x$. By the definition of φ, $u_{zero}\langle prog(x)\rangle_T u$ and $u_{zero}\langle prog(x)\rangle_T v$. By *Lemma 3.4*, $u = v$. To show that φ is a surjection, let $x \in N^5$. By *Lemma 3.3*, there is a u such that $u_{zero}\langle prog(x)\rangle_T u$, so $\varphi(u) = x$. ∎

Finally, we will show that φ makes T a grid, by proving that the three defining properties of grids hold of T and φ.

(1) Suppose $u\langle A_i\rangle_T v$. Then $u_{zero}\langle prog(\varphi(u))\rangle_T u$ and $u_{zero}\langle prog(\varphi(u));A_i\rangle_T v$. By *Lemma 3.7*, $u_{zero}\langle prog(\sigma_i(\varphi(u)))\rangle_T v$. By *Lemma 3.13*, $\varphi(v) = \sigma_i(\varphi(u))$.

Conversely, suppose $\varphi(v) = \sigma_i(\varphi(u))$. Then $u_{zero}\langle prog(\varphi(u))\rangle_T u$ and $u_{zero}\langle prog(\sigma_i(\varphi(u)))\rangle_T v$. By *Lemma 3.7*, $u_{zero}\langle prog(\varphi(u));A_i\rangle_T v$. By *Lemma 3.5*, $u\langle A_i\rangle_T v$.

(2) Without loss of generality let $i = 1$. Suppose $u\langle B_1\rangle_T v$ where $\varphi(u) = x$ and $\varphi(v) = y$. Then $u_{zero}\langle prog(x);B_1\rangle_T v$. By *Lemma 3.8*, $u_0\langle A_1^{x_1};B_1;A_2^{x_2};A_3^{x_3};A_4^{x_4};A_5^{x_5}\rangle_T v$. By $axiom_3$, $u_{zero} \models [B_1]false$, so $x_1 > 0$. By *Lemma 3.6*, $u_{zero}\langle A_1^{x_1-1};A_2^{x_2};A_3^{x_3};A_4^{x_4};A_5^{x_5}\rangle_T v$. Therefore $x = \varphi(u) = \sigma_1(\varphi(v)) = \sigma_1(y)$. Conversely, suppose $\varphi(u) = \sigma_1(\varphi(v)) = \sigma_1(x)$. Then $u_{zero}\langle prog(\sigma_1(x))\rangle_T u$ and $u_{zero}\langle prog(x)\rangle_T v$. By *Lemma 3.6*, $u_{zero}\langle A_1^{x_1+1};B_1;A_2^{x_2};A_3^{x_3};A_4^{x_4};A_5^{x_5}\rangle_T v$. By *Lemma 3.8*, $u_{zero}\langle prog(\sigma_1(x));B_1\rangle_T v$. By *Lemma 3.5*, $u\langle B_1\rangle_T v$.

(3) Suppose $u \models_T R$ and $\varphi(u)_1 = \varphi(v)_1$. Let $\varphi(u) = x$, $\varphi(v) = y$. Then $u_{zero}\langle prog(x)\rangle_T u$ and $u_{zero}\langle A_1^{x_1};A_2^{y_2};A_3^{y_3};A_4^{y_4};A_5^{y_5}\rangle_T v$. By *Lemmas 3.6* and *3.8*, $u_{zero}\langle prog(x);B_2^{x_2};B_3^{x_3};B_4^{x_4};B_5^{x_5};A_2^{y_2};A_3^{y_3};A_4^{y_4};A_5^{y_5}\rangle_T v$. By *Lemma 3.5*, $u\langle B_2^{x_2};B_3^{x_3};B_4^{x_4};B_5^{x_5};A_2^{y_2};A_3^{y_3};A_4^{y_4};A_5^{y_5}\rangle_T v$. By *Lemma 3.9*, $v \models_T R$. This completes the proof of *Proposition 3.1*. ∎

Corollary 3.14: If α contains all primitive programs appearing in a formula p, then p is valid in all grids if and only if $grid\text{-}scheme_Q^{PDL}$ infers p.

Proof: By definition, $grid\text{-}scheme_Q^{PDL}$ infers p if and only if p is valid in all structures in which $grid\text{-}scheme_Q^{PDL}$ is valid. By *Lemma 2.1*, the latter is true if and only if p is valid in all *PDL*-collapses of structures in which $grid\text{-}scheme_Q^{PDL}$ is valid. By *Corollary 2.3*, this is so if and only if p is valid in all α-cuts of *PDL*-collapses of structures in which $grid\text{-}scheme_Q^{PDL}$ is valid. By *Proposition 3.1*, this is so if and only if p is valid in all grids. ∎

Notation: Let α^* abbreviate $(A_1 \cup A_2 \cup A_3 \cup A_4 \cup A_5 \cup B_1 \cup B_2 \cup B_3 \cup B_4 \cup B_5)^*$.

Corollary 3.15: If p is a formula all of whose atomic programs are in α, then p is valid in all grids if and only if $([\alpha^*]grid\text{-}scheme)_Q^{PDL}$ implies p.

Proof: Left to the reader. ∎

4 Π_1^1-completeness of the Deducibility Problem for *PDL*

Lemma 4.1: Let $f: 2^N \times N^3 \to N$ be a partial recursive function of one set variable and three integer variables. There is a *PDL* program a_f such that, in every grid S, $u\langle a_f\rangle_S v$ if and only if $[\varphi(v)]_1 = f(X_S, \varphi(u)_1, \varphi(u)_2, \varphi(u)_3)$, where $X_S = \{\varphi(w)_1 \mid w \models_S R\}$.

Proof: An oracle counter machine is a computing device possessing registers capable of holding arbitrary nonnegative integers and a processor capable of incrementing and decrementing (when the result is nonnegative) the contents of a specified register, testing whether the contents of a specified register is zero or not, and testing the contents of the first register for membership in a fixed but arbitrary set called the "oracle". (The formal definition is analogous to that of oracle Turing machines [5, 6] and is omitted.) A 5-counter machine is capable of computing any partial recursive function of one set variable and three integer variables, where we assume that the three inputs are initially stored in the first three registers (the extra two registers are for temporary results and may initially contain arbitrary values) and that the single integer output is stored, at the end, in the first register. A program a_f to compute such a function f can be written as a regular program using the primitives (where $1 \leq i$

≤ 5): A_i to increment register i, B_i to decrement register i, $zero_i?$ and $\neg zero_i?$ to test register i for zero, and $R?$ and $\neg R?$ to test whether the contents of register 1 is in the oracle set X_S. In a grid S the standard *PDL* semantics interprets a_f as a program which computes f, i.e. that $u\langle a_f\rangle_S v$ if and only if $\varphi(v)_1 = f(X_S, \varphi(u)_1, \varphi(u)_2, \varphi(u)_3)$. ∎

For the remainder of this paper let Y be a fixed Π_1^1-complete set of natural numbers, so that there is a fixed recursive function $f(X, x, y, z)$ of one set variable and three integer variables such that $Y = \{x \mid \forall X \subseteq N.\ \exists y.\ \forall z.\ f(X, x, y, z) = 0\}$.

Corollary 4.2: There is a *PDL* formula p_Y such that for all natural numbers m, the formula $zero_1 \rightarrow \langle A_1{}^m\rangle p_Y$ is valid in all grids if and only if $m \in Y$.

Proof: By the preceding lemma, for all grids S and states u, $u \models_S \langle a_f\rangle zero_1$ if and only if $f(X_S, \varphi(u)_1, \varphi(u)_2, \varphi(u)_3) = 0$. The program $B_i{}^*;A_i{}^*$ is capable of arbitrarily altering the contents of the i^{th} register. Hence $u \models_S [B_3{}^*;A_3{}^*]\langle a_f\rangle zero_1$ if and only if $\forall z \in N.\ f(X_S, \varphi(u)_1, \varphi(u)_2, z) = 0$. Similarly, $u \models_S \langle B_2{}^*;A_2{}^*\rangle[B_3{}^*;A_3{}^*]\langle a_f\rangle zero_1$ if and only if $\exists y \in N.\ \forall z \in N.\ f(X_S, \varphi(u)_1, y, z) = 0$. Let p_Y be $\langle B_2{}^*;A_2{}^*\rangle[B_3{}^*;A_3{}^*]\langle a_f\rangle zero_1$. If $u \models_S zero_1$, then $u \models_S \langle A_1{}^m\rangle p_Y$ if and only if $\exists y \in N.\ \forall z \in N.\ f(X_S, m, y, z) = 0$. As S ranges over all grids, X_S ranges over all sets of nonnegative integers. Therefore, $zero_1 \rightarrow \langle A_1{}^m\rangle p_Y$ is valid in all grids if and only if $\forall X \subset N.\ \exists y \in N.\ \forall z \in N.\ f(X, m, y, z) = 0$, i.e. if and only if $m \in Y$. ∎

Proposition 4.3: The scheme inference (respectively, implication) problem for *PDL* is Π_1^1-complete.

Proof: By *Corollaries 3.14 (3.15)* and *4.2*, there is a *PDL* formula p_Y such that $m \in Y$ if and only if $grid\text{-}scheme_Q{}^{PDL}$ (respectively, $([\alpha^*]grid\text{-}scheme)_Q{}^{PDL}$) infers (implies) $zero_1 \rightarrow \langle A_1{}^m\rangle p_Y$. This proves that Π_1^1 is many-one reducible to the scheme inference (implication) problem for *PDL*. It is not hard to show that either problem is in Π_1^1; we omit the proof. ∎

We now define some sublanguages of *PDL* and show that the scheme implication and inference problems are Π_1^1-complete for some of these sublanguages.

Definition: The formulae of *test-free propositional dynamic logic* are those in which no tests appear.

Theorem 4.4: If L is a subset of *PDL* which contains *test-free-PDL*, then the scheme inference (respectively, implication) problem for L is Π_1^1-complete.

Proof: The tests of p_Y are of the form $zero_i?$, $\neg zero_i?$, $R?$, and $\neg R?$. Choose new atomic programs $C_1, \ldots, C_{12}$. Let q_Y be the result of substituting C_1 for $zero_1?, \ldots, C_{12}$ for $\neg R?$ in p_Y. Let *free-scheme* be *grid-scheme* $\wedge$ $(\bigwedge_{1\leq i\leq 12} Q \rightarrow [C_i]Q) \wedge zero_1 \leftrightarrow \langle C_i\rangle true \wedge \ldots \wedge \neg R \leftrightarrow \langle C_{12}\rangle true)$. We leave it to the reader to show that the problem of deciding, for a given m, whether or not $free\text{-}scheme_Q{}^L$ (respectively, $([\alpha^*]free\text{-}scheme)_Q{}^L$) infers (implies) $zero_1 \rightarrow \langle A_1{}^m\rangle q_Y$ is Π_1^1-complete. ∎

Definition: The set of programs, Π_d, and the set of formulae, Φ_d, of *deterministic propositional dynamic logic* (*DPDL*) are defined inductively as follows.

Π_d: (1) $\Pi_0 \subseteq \Pi_d$ and $\theta, \lambda \in \Pi_d$

(2) If a, $b \in \Pi_d$ and $p \in \Phi_d$, then $(a;b)$, (*if p then a else b*), (*while p do a*) $\in \Pi_d$

Φ_d: (1) $\Phi_0 \subseteq \Phi_d$

(2) If p, $q \in \Phi_d$ then $\neg p$, $p\&q \in \Phi_d$

(3) If $a \in \Pi_d$ and $p \in \Phi_d$ then $\langle a\rangle p \in \Phi_d$

Definition: The formulae of *atomic-test-DPDL* are those formulae of *DPDL* in which the constructions *if p then a else b* and *while p do a* appear only when p is an atomic proposition.

Theorem 4.5: If L is a subset of *PDL* which contains *atomic-test-DPDL*, then the scheme inference problem for L is Π_1^1-complete.

Proof: First, note that a_f of *Lemma 4.1* can easily be written as a program in Π_d. Second, note that for all programs a and formulae p, $\langle a^*\rangle p$ is equivalent to $\langle$*while* $\neg p$ *do* $a\rangle$*true*. Hence, there is a formula r_Y in Π_d which is equivalent to $p_Y =_{df} \langle B_2^*;A_2^*\rangle[B_3^*;A_3^*]\langle a_f\rangle zero_1$. Finally, note that every conjunct of *grid-scheme* is in Π_d except for *zero-axiom* $=_{df} \langle B_1^*;B_2^*;B_3^*;B_4^*;B_5^*\rangle zero$. There is a formula in Π_d which is equivalent to *zero-axiom* in all structures; let *det-scheme* be *grid-scheme* with *zero-axiom* replaced by this formula. Let the non-atomic tests of *det-scheme* and r_Y be $r_1?, \ldots, r_n?$. A maximal occurrence of a test $r_i?$ in a formula q is an occurrence which is not itself contained in an occurrence of another test $r_j?$ occurring in q. Choose new atomic propositions $T_1, \ldots, T_k$. Let *new-scheme* and s_Y be the result of substituting, for $1 \leq i \leq k$, $T_i?$ for each maximal occurrence of $r_i?$ in *det-scheme* and r_Y, and for $1 \leq i \leq k$, let t_i be the result of substituting, for $j \neq i$, $T_j?$ for each maximal occurrence of $r_j?$ in r_i. Let *equiv-scheme* be *new-scheme* $\wedge$ $(\bigwedge_{1\leq i\leq k} T_i \leftrightarrow t_i)$. We leave it to the reader to show that the problem of deciding, for a given m, whether or not *equiv-scheme*$_Q^L$ infers $\langle A_1^m\rangle s_Y$ is Π_1^1-complete. ∎

5 Conclusions and Open Problems

Because of its many decidable properties, *PDL* appears to be a reasonably tractable extension of propositional logic. However, we have revealed a dramatic contrast between *PDL* and ordinary propositional logic in the case of the scheme deducibility problem, which is Π_1^1-complete for *PDL*, but decidable for propositional logic.

An important hint at the power of *PDL* axiom schemes was provided by the observation of Salwicki and Pratt [2], who showed that the nonnegative integers could be characterized (as cuts of *PDL*-collapsed structures) by a finite set of axiom schemes. Hence this set of axiom schemes does not satisfy the finite model property, namely these schemes have a model but no finite model. Since all the previously known decidability results for *PDL* ultimately rest on the finite model property of *PDL* formulae, the Salwicki-Pratt observation helps clarify the contrast between schemes and finite sets of axioms.

However, violation of the finite model property should not be taken as *prima facie* evidence of undecidability. For example, Mirkowska has observed that the nonnegative integers can also be uniquely characterized by a single formula of *PDL* extended with a looping predicate and the converse operation on programs [3]. Nevertheless, Streett has shown that this extension of *PDL* is still decidable (in fact, elementary recursive) [7].

The degrees of undecidability (or decidability) of several restricted deducibility problems remain open questions.

Open Problem: Is the scheme implication problem for *DPDL* or *atomic-test-DPDL* Π_1^1-complete?

Open Problem: How hard are the scheme deducibility problems for propositional temporal and modal logics?

Acknowledgement: We are grateful to A. Salwicki for pointing out the possibility of characterizing the integers by *PDL* axiom schemes, and for several useful discussions about these results. David Harel is partially responsible for the proof of *Theorem 4.4.*

References

1. Fischer, M. J. and R. E. Ladner, "Propositional Dynamic Logic of Regular Programs", *JCSS, 18,* 194-211, 1979.

2. Mirkowska, G., "Complete Axiomatization of Algorithmic Properties of Program Schemes with Bounded Nondeterministic Interpretations", *Proceedings of the 12th ACM Symposium on Theory of Computing,* 14-21, 1980.

3. Mirkowska, G., personal communication. Warsaw, July, 1980.

4. Pratt, V. R., "Models of Program Logics", *JCSS, 20,* 231-254, 1980.

5. Rogers, H., *Theory of Recursive Functions and Effective Computability,* McGraw-Hill Book Company, 1967.

6. Shoenfield, J. R., *Mathematical Logic,* Addison-Wesley Publishing Company, 1967.

7. Streett, R. S., "Propositional Dynamic Logic of Looping and Converse", *Proceedings of the 13th ACM Symposium on Theory of Computing,* 1981.

---

*This research was supported in part by the National Science Foundation, grant nos. MCS 7719754, MCS 8010707, and MCS 7910261, and by a grant to the MIT Laboratory for Computer Science by the IBM Corporation.