MIT/LCS/TM-190

DETERMINISTIC PROPOSITIONAL DYNAMIC LOGIC: FINITE MODELS, COMPLEXITY, AND COMPLETENESS

Mordechai Ben-Ari
Joseph Y. Halpern
Amir Pnueli

January 1981

Deterministic Propositional Dynamic Logic: Finite Models, Complexity, and Completeness

Mordechai Ben-Ari,
Department of Mathematical Sciences,
Division of Computer Sciences,
Tel Aviv University,
Ramat Aviv, Israel.

Joseph Y. Halpern,
Mathematics Department,
Harvard University,
Cambridge, Massachusetts 02138

Amir Pnueli,
Department of Mathematical Sciences,
Division of Computer Sciences,
Tel Aviv University,
Ramat Aviv, Israel.

December 17, 1980.

Abstract: Let p be a formula in deterministic propositional dynamic logic. A decision procedure for the satisfiability of p is given along with a construction of a finite model for every satisfiable p. The decision procedure runs in deterministic time 2^{cn} and the size of the model is bounded by $n^{2 \cdot 4^n}$, where n is the length of p. Finally, a complete axiomatization for deterministic propositional dynamic logic is given, based on the Segerberg axioms for propositional dynamic logic.

KEYWORDS: deterministic propositional dynamic logic, propositional dynamic logic, decision procedure, tableau method, completeness.

This research was partially supported by grants from the National Science and Engineering Research Council of Canada, NSF Grant #MCS 7719754, the Israeli Academy of Sciences and Humanities, and the Basic Research Foundation.

1. Introduction

Dynamic logic, an outgrowth of modal logic, was introduced by Pratt [5] as a logical theory capable of expressing properties of computer programs. Fischer and Ladner [1] have investigated the purely logical properties of the propositional fragment of dynamic logic (PDL). Their principal results are a decision procedure for satisfiability and a proof of the finite model property: if a formula in PDL is satisfiable then it is satisfiable a finite model, in fact one of size 2^n. These results were re-derived and extended by Pratt [6, 7] who gave a 2^{cn} deterministic time algorithm for PDL using tableau techniques. Segerberg [8] proposed an axiomatization for PDL, which was later shown to be complete by various researchers (see [2] for an elementary proof and further references).

Deterministic PDL (DPDL) is the logical theory with the same syntax as PDL but with its semantics restricted so that in each state an atomic program specifies at most on successor state. Parikh [3] has given a decision procedure for DPDL as a corollary to the decision procedure for a very strong theory: second order process logic. However, that procedure is of non-elementary complexity and cannot be considered practical for DPDL.

We give a 2^{cn} deterministic time decision procedure for satisfiability in DPDL. This agrees with the lower bound shown by Parikh [4]. The proof uses the notion of a *partial D model* for a formula p, which is precisely what we end up with when we apply the Fischer-Ladner factor model construction to a DPDL model for p.

We introduce the syntax and semantics of PDL and DPDL in section 2. In section 3 we review the ideas of the Fischer-Ladner proof of the finite model property for PDL, and provide the motivation for and definitions of partial PDL, DPDL, and D models for a formula p. In section 4 we prove the main technical result, namely that a formula of size n is DPDL satisfiable iff it has a partial D model of size 2^n iff it has a DPDL model of size $n^2 \cdot 4^n$. We use this result in section 5 to give us the decision procedure. It is worth noting that we do *not* have to construct a DPDL model for p in order to decide whether or not p is DPDL satisfiable. Finally, in section 6, we use the methods of [2] to give a complete Segerberg-like axiomatization of DPDL.

Valiev has sketched a completeness proof for DPDL in [9] and a decision procedure in [10]. He suggests that the techniques of [10] can give a finite model but does not give details.

2. Syntax and Semantics

2.1 *Syntax*: The alphabet for PDL (as well as DPDL), $\mathcal{L}$, consists of a set Φ_0, whose elements are called atomic formulas, a set Σ_0, whose elements are called atomic programs, and the symbols $\cup$, ;, $*$, ?, $\neg$, $<$, $>$, (,).

The set of programs, Σ, and the set of formulas, Φ, are defined inductively using the following rules:

1. any atomic program in Σ_0 is a program;
2. if a and b are programs, then so are (a;b), ($a \cup b$), and a^*;
3. any atomic formula in Φ_0 is a formula;
4. if p is a formula and a is a program, then $\neg p$ and $\langle a \rangle p$ are formulas;
5. If p is a formula, then p? is a program.

We also use the following abbreviations: $p \wedge q$ for $\langle p? \rangle q$, $p \vee q$ for $\neg(\neg p \wedge \neg q)$, $p \rightarrow q$ for $\neg p \vee q$, $p \equiv q$ for $(p \rightarrow q) \wedge (q \rightarrow p)$, and $[a]p$ for $\neg\langle a \rangle \neg p$.

The *length* of a formula p, written $|p|$, is the length of p regarded as a string over $\mathcal{L}$.

2.2 *Notation*: We will normally reserve P, Q, R, ... for members of Φ_0, and A, B, C, ... for members of Σ_0. The letters p, q, r, ... denote formulas, while the letters a, b, c, ... denote programs.

2.3 *Definition*: A *PDL structure* M is a triple (S, π, ρ) where S is a set whose elements are called *states*, $\pi: \Phi \rightarrow \mathcal{P}(S)$ is an assignment of formulas to sets of states, and $\rho: \Sigma \rightarrow \mathcal{P}(S \times S)$ is a mapping of programs into binary relations on S which satisfies the following constraints:

1. $\rho(a;b) = \rho(a) \circ \rho(b)$ (composition of relations)
2. $\rho(a \cup b) = \rho(a) \cup \rho(b)$ (union of relations)
3. $\rho(a^*) = (\rho(a))^*$ (reflexive and transitive closure)
4. $\rho(p?) = \{(s, s) \mid p \in \pi(s)\}$

A *DPDL structure* satisfies in addition:

5. For all $A \in \Sigma_0$, $\rho(A)$ defines a partial function;
   i.e. if $(s, t), (s, t') \in \rho(A)$, then $t = t'$.

If $p \in \Phi$, then we can view $\pi(p)$ as the set of states in which p is true. And if $a \in \Sigma$, then $\rho(a)$ is the input-output relation of program a, i.e., $(u, v) \in \rho(a)$ means that by starting in state u and running program a we can halt in state v.

The *size* of a structure $M = (S, \pi, \rho)$ is the cardinality of S.

2.4 *Definition*: A *(D)PDL model* is a (D)PDL structure (S, π, ρ) satisfying the following additional constraints on π:

6. $\pi(\neg p) = S - \pi(p)$
7. $\pi(\langle a \rangle p) = \{s \in S \mid \exists t((s, t) \in \rho(a) \text{ and } t \in \pi(p))\}$

2.5 *Remarks*: 1. Given $\pi' : \Phi_0 \to \mathcal{P}(S)$, $\rho' : \Sigma_0 \to \mathcal{P}(S \times S)$, we can always uniquely extend π' to $\pi : \Phi \to \mathcal{P}(S)$ and ρ' to $\rho : \Sigma \to \mathcal{P}(S \times S)$ so that conditions 1-4, 6, and 7 hold. Moreover, if ρ' satisfies condition 5, then so does ρ. Thus, for a (D)PDL model, π and ρ are completely defined by their actions on the primitive formulas and programs.

2. We will say *t is an a-successor of s* in a structure if $(s, t) \in \rho(a)$. In a DPDL model, each $s \in S$ has at most one A-successor for all $A \in \Sigma_0$. Any (D)PDL model $M = (S, \pi, \rho)$ can be viewed as a directed graph, with the nodes labelled by states in S. We join s to t by an edge labelled A iff $(s, t) \in \rho(A)$. The graph together with π uniquely defines M.

2.6 *Definitions*: Let $M = (S, \pi, \rho)$. Then

1. $M, s \models p$ (p is *true* in $s \in S$) iff $p \in \pi(s)$,
2. $M \models p$ (p is *satisfiable* in M) iff, for some $s \in S$, we have $M, s \models p$,
3. a formula p is *(D)PDL satisfiable* iff for some (D)PDL model M, $M \models p$,
4. $\models_{(D)} p$ (p is *(D)PDL valid*) iff for all (D)PDL models $M = (S, \pi, \rho)$ and all $s \in S$, we have $M, s \models p$.

2.7 *Lemma*:

1. If $\models p$, then $\models_D p$. (It thus follows that if p is DPDL satisfiable then p is PDL satisfiable.)
2. $M, s \models \langle p? \rangle q$ iff $M, s \models p$ and $M, s \models q$. (This justifies the abbreviation $p \wedge q$ for $\langle p? \rangle q$).
3. $\models \langle a;b \rangle p \equiv \langle a \rangle \langle b \rangle p$.
4. $\models \langle a \cup b \rangle p \equiv \langle a \rangle p \vee \langle b \rangle p$.
5. $\models \langle a^* \rangle p \equiv p \vee \langle a \rangle \langle a^* \rangle p$.
6. For $A \in \Sigma_0$, $\models_D \langle A \rangle p \to [A]p$.

Proof: Straightforward from the definitions. In 1, note that a DPDL model is *a fortiori* a PDL model.

2.8 *Remark*: Note that the converse to Lemma 2.7(1) fails. For example, $\langle A\rangle p \wedge \langle A\rangle \neg p$ is PDL satisfiable but not DPDL satisfiable, while its negation is DPDL valid but not PDL valid.

3. FL-Closure and Partial Models

3.1 The Fischer-Ladner closure of a formula p_0, $FL(p_0)$, is defined to be the least set F such that $p_0 \in F$ and

1.	$\neg p \in F$	$\rightarrow$	$p \in F$
2.	$\langle a\rangle p \in F$	$\rightarrow$	$p \in F$
3.	$\langle a;b\rangle p \in F$	$\rightarrow$	$\langle a\rangle\langle b\rangle p \in F$
4.	$\langle a \cup b\rangle p \in F$	$\rightarrow$	$\langle a\rangle p, \langle b\rangle p \in F$
5.	$\langle a^*\rangle p \in F$	$\rightarrow$	$\langle a\rangle\langle a^*\rangle p \in F$
6.	$\langle p?\rangle q \in F$	$\rightarrow$	$p, q \in F$

3.2 *Theorem*: (Fischer-Ladner) If $|p_0| = n$, then $|FL(p_0)| \leq n$.

Proof: See [1]. ∎

3.3 *Definition*: If p_0 is a formula, let $\Sigma_0(p_0) = \{A \in \Sigma_0 \mid A \text{ appears in } p_0\}$. Let $\Sigma(p_0)$ be the least set containing $\Sigma_0(p_0)$ such that if $a, b \in \Sigma(p_0)$ so are $a \cup b$, $a;b$, and a^*, and if $q \in FL(p_0)$, $q? \in \Sigma(p_0)$.

The point of $FL(p_0)$ and $\Sigma(p_0)$ is that if we want to construct a PDL model satisfying p_0, the only formulas and programs which we must take into account are those in $FL(p_0)$ and $\Sigma(p_0)$. This comment is made more precise in the proof of the following theorem.

3.4 *Theorem* (Fischer-Ladner): If $|p_0| = n$, then p_0 is PDL satisfiable iff p_0 is satisfiable in a PDL model of size $\leq 2^n$.

Proof: We just present a sketch here. The reader is referred to [1] for more details. Suppose $M = (S, \pi, \rho)$ is a PDL model satisfying p_0. Define an equivalence relation $\equiv$ on S via

$s_1 \equiv s_2$ iff (M, $s_1 \models p$ iff M, $s_2 \models p$ for all $p \in FL(p_0)$).

Since an equivalence class is completely determined by which of the n formulas in $FL(p_0)$ it satisfies, there are at most 2^n equivalence classes.

Let $[s] = \{s' \in S | s' \equiv s\}$, and let $S' = \{[s] | s \in S\}$. Note $|S'| \leq 2^n$.

Define $\pi'': \Phi_0 \to \mathcal{P}(S')$, $\rho'': \Sigma_0 \to \mathcal{P}(S' \times S')$ via

$\pi''(P) = \{[s] | s \in \pi(P)\}$

$\rho''(A) = \{([s], [t]) | (s, t) \in \rho(A)\}$

Extend π'' to $\pi': \Phi \to \mathcal{P}(S')$, ρ'' to $\rho': \Sigma \to \mathcal{P}(S' \times S')$ to get a PDL models. Let $M' = (S', \pi', \rho')$. Then it can be shown that for $p \in FL(p_0)$,

M, $s \models p$ iff M', $[s] \models p$. ∎

3.5 We would like to apply the above ideas to showing that a formula is DPDL satisfiable iff it has a finite model. However, when we try to carry out the above construction starting with a DPDL model $M = (S, \pi, \rho)$ satisfying p_0, we find that in general $M' = (S', \pi', \rho')$ is not a DPDL model. What goes wrong is that there might be states $s_1, s_2, t_1, t_2 \in S$ with $(s_1, t_1) \in \rho(A)$, $(s_2, t_2) \in \rho(A)$, $s_1 \equiv s_2$, but $t_1 \not\equiv t_2$. Thus both $([s_1], [t_1]), ([s_1], [t_2]) \in \rho'(A)$, so $\rho'(A)$ does not define a partial function.

However, the M' so constructed does have one important property, namely:

if $\langle A \rangle p \in FL(p_0)$ and M', $[s'] \models \langle A \rangle p$, then for all $[t']$ such that $([s'], [t']) \in \rho'(A)$, M', $[t'] \models p$.

To see this, suppose M', $[s'] \models \langle A \rangle p$ and for some $[t']$ with $([s'], [t']) \in \rho'(A)$ we have M', $[t'] \models \neg p$. Then, by definition of ρ', there exists $s \in [s']$, $t \in [t']$ with $(s, t) \in \rho(A)$. Moreover, M, $t \models \neg p$ and M, $s \models \langle A \rangle p$. But since M is a DPDL model, t is the *unique* A-successor of s in M, so M, $t \models p$, contradicting M, $t \models \neg p$.

The difference between this property and that of Lemma 2.7.6: $\models_D \langle A \rangle p \to [A]p$ is that the property is required to hold only for $\langle A \rangle p \in FL(p_0)$ and not for *all* $\langle A \rangle p$ in the language.

The above comments motivate the following definition of partial model. The idea is that a partial model for p_0 should be a structure which obeys the conditions required of a model for p_0, at least for the formulas appearing in and $FL(p_0)$. More formally we have

3.6 *Definition*: A *partial (D)PDL model* for p_0 is a (D)PDL structure $M = (S, \pi, \rho)$ such that $\pi(p_0) \neq \varnothing$ and

6′. For $\neg q \in FL(p_0)$, $\pi(\neg q) = S - \pi(q)$,

7′. For $\langle a \rangle q \in FL(p_0)$, $\pi(\langle a \rangle q) = \{s \mid \exists t((s, t) \in \rho(a) \text{ and } t \in \pi(p))\}$.

A *partial D model* for p_0 satisfies 1-4, 6′, 7′ and

8′. For $\langle A \rangle q \in FL(p_0)$,
if $s \in \pi(\langle A \rangle q)$ then for all t such that $(s, t) \in \rho(A)$, $t \in \pi(q)$.

Note that a partial DPDL model for p_0 is trivially a partial D model for p_0.

The following lemma is just a refinement of Lemma 1 in [6]:

3.7 *Lemma*: A formula p_0 is (D)PDL satisfiable in a model of size N iff there is a partial (D)PDL model for p_0 of size N.

Proof: We consider the PDL case; the DPDL case is exactly the same. It is clear that any PDL model satisfying p_0 is automatically a partial PDL model for p_0. For the converse, suppose $M = (S, \pi, \rho)$ is a partial PDL model for p_0. Let $\pi'' = \pi|\Phi_0$, $\rho'' = \rho|\Sigma_0$, and extend π'' and ρ'' to mappings $t\pi'$: $\Phi \to \mathscr{P}(S)$ and ρ': $\Sigma \to \mathscr{P}(S \times S)$ which satisfy the PDL model constraints. Then it is easy to show by induction on the structure of formulas and programs that

$$\rho'|\Sigma(p_0) = \rho|\Sigma(p_0),\ \pi'|FL(p_0) = \pi|FL(p_0).$$

Thus $M' = (S, \pi', \rho')$ is a PDL model for p_0. ∎

We conclude from this lemma that

p_0 is DPDL satisfiable
↔ there is a partial DPDL model for p_0
→ there is a partial D model for p_0.

We will show that the second implication is actually an equivalence.

4. Constructing a Partial DPDL Model from a Partial D Model

We are now ready to state our major theorem:

4.1 *Theorem*: Let $|p_0| = n$. Then the following are equivalent:

(a) p_0 is DPDL satisfiable,

(b) there is a partial D model for p_0 of size $\leq 2^n$,

(c) there is a partial DPDL model for p_0 of size $\leq n^2 \cdot 4^n$.

Proof: (a) → (b) follows immediately from the Fischer-Ladner construction presented in Theorem 3.4 and the comments in 3.5.

(c) → (a) follows immediately from Lemma 3.7.

(b) → (c) will require a little more work. First we need some definitions and lemmas.

4.2 *Definition*: For $a \in \Sigma$, we define $\tau(a)$, the set of *a-trajectories* in $M = (S, \pi, \rho)$ by induction on the structure of a (cf. [6, p.328]):

1. $\tau(A) = \rho(A)$,
2. $\tau(a \cup b) = \tau(a) \cup \tau(b)$,
3. $\tau(a;b) = \tau(a) \circ \tau(b)$
   $= \{(s, \ldots, u, \ldots, t) \mid (s, \ldots, u) \in \tau(a) \text{ and } (u, \ldots, t) \in \tau(b)\}$,
4. $\tau(a^*) = \{(s) \mid s \in S\} \cup (\cup_{i \geq 1} \tau(a^i))\}$,
5. $\tau(p?) = \{(s) \mid M, s \models p)\}$.

The *length* of the trajectory $(s_0, \ldots, s_k)$ is k.

Note that $(s, t) \in \rho(a)$ iff there exists an a-trajectory $(s_0, \ldots, s_k)$ with $s = s_0$ and $t = s_k$. Such a trajectory is called an *a-trajectory from s to t*. Informally, an a-trajectory from s to t describes the path taken by a in getting from the node labelled s to the node labelled t in the graph corresponding to the structure M.

For the balance of this section, let $M = (S, \pi, \rho)$ be a partial PDL model for p_0. The following lemma shows that the structure of a trajectory as a sequence of states joined by atomic programs is reflected in the elements of the FL closure in each state.

4.3 *Lemma*: Suppose $\langle a_1 \rangle \ldots \langle a_h \rangle p \in FL(p_0)$, $M, s_0 \models \langle a_1 \rangle \ldots \langle a_h \rangle p$ and $M, s_k \models p$, and $(s_0, \ldots, s_k)$ is an $a_1;\ldots;a_h$ trajectory of length > 0.

Then for all $i < k$, there exist $A \in \Sigma_0(p_0)$, $b_1, \ldots, b_m \in \Sigma(p_0)$ $(m \geq 0)$, such that

(a) $\langle A\rangle\langle b_1\rangle\ldots\langle b_m\rangle p \in FL(p_0)$ (and hence $\langle b_1\rangle\ldots\langle b_m\rangle p \in FL(p_0)$),

(b) $(s_i, s_{i+1}, \ldots, s_k) \in \tau(A;b_1;\ldots;b_m)$ (and hence $M, s_i \models \langle A\rangle\langle b_1\rangle\ldots\langle b_m\rangle p$ and $M, s_{i+1} \models \langle b_1\rangle\ldots\langle b_m\rangle p$),

(c) $(s_0, \ldots, s_i) \circ \tau(A;b_1;\ldots;b_m) \subseteq \tau(a_1;\ldots;a_h)$ (and hence $(s_0, s_i) \circ \rho(A;b_1;\ldots;b_m) \subseteq \rho(a_1;\ldots;a_h)$).

Proof: By a straightforward induction on h, i and the structure of a_1. ∎

4.4 *Definition*: If $M, s \models q$, where $q = \langle a_1\rangle\ldots\langle a_h\rangle p$ and p is not of the form $\langle c\rangle r$, then q is *fulfilled for s by t* if $(s, t) \in \rho(a_1;\ldots;a_h)$ and $M, t \models p$. We say q is *immediately fulfilled by s* if $h = 0$ or if $(s) \in \tau(a_1;\ldots;a_h)$ and $M, s \models p$. $\langle A\rangle\langle b_1\rangle\ldots\langle b_m\rangle p$ is a *derivative* of q for s at t if $M, t \models \langle A\rangle\langle b_1\rangle\ldots\langle b_m\rangle p$ and $(s, t) \circ \rho(A;b_1;\ldots;b_m) \subseteq \rho(a_1;\ldots;a_h)$. Note that if q′ is a derivative of q for s at t, and if q″ is a derivative of q′ for t at u, then q″ is a derivative of q for s at u. Thus the derivative possesses a kind of transitivity property. Moreover, it follows from the definition that if q′ is a derivative of q for s at t, and q′ is fulfilled for t by u, then q is fulfilled for s by u. Informally this says that if q′ is a derivative of q for s at t, then t is a way station on a trajectory to fulfilling q for s. ∎

4.5 *Lemma*: If M is of size N, $|p_0| = n$, $\langle a_1\rangle\ldots\langle a_h\rangle p \in FL(p_0)$, $M, s \models \langle a_1\rangle\ldots\langle a_h\rangle p$, $M, t \models p$, and $(s, t) \in \rho(a_1;\ldots;a_h)$, then there exists an $a_1;\ldots;a_h$-trajectory from s to t of length $\leq nN$.

Proof: Suppose $(s_0, \ldots, s_k)$ is the shortest $a_1;\ldots;a_h$ trajectory from s to t and $k > nN$. By Lemma 4.3, with each s_i, $i < k$, we can associate a derivative $q_i \in FL(p_0)$ of the form $\langle A\rangle\langle b_1\rangle\ldots\langle b_m\rangle p$ such that $(s_i, \ldots, s_k) \in \tau(A;b_1;\ldots;b_m)$ and $(s_0, \ldots, s_i) \circ \tau(A;b_1;\ldots;b_m) \subseteq \tau(a_1;\ldots;a_h)$. There are at most n distinct q_i′s (since $|FL(p_0)| \leq n$) and N distinct s_i′s. Since $k > nN$, we must have $(s_i, q_i) = (s_j, q_j)$ for some $i < j$. But then it is easily checked that $(s_0, \ldots, s_i, s_{j+1}, \ldots, s_k)$ is an $a_1;\ldots;a_h$-trajectory from s to t, contradicting the assumption that $(s_0, \ldots, s_k)$ was the shortest such trajectory. ∎

Proof of Theorem 4.1

Let M be a partial D model for p_0 of size $\leq 2^n$. We would like to construct a partial DPDL model for p_0 from M. We will in fact construct a tree-like partial DPDL model in stages. At the root we will put s_0, where $s_0 \in S$ such that $M, s_0 \models p_0$. Then we will have to ensure that for each formula $\langle a\rangle p \in FL(p_0)$ such that $M, s_0 \models \langle a\rangle p$ we add an a-trajectory leading to some node $M, t \models p$. We must also do this in a deterministic way, i.e. for each $A \in \Sigma(p_0)$ and each node t on the tree, there should only be one A-successor of t. Then for every new node that we add we must also ensure that every formula of the form $\langle a\rangle p$ true at that node is eventually fulfilled.

For each $s \in S$, let $D(s) = \{\langle A\rangle p \in FL(p_0) \mid M, s \models \langle A\rangle p\}$.

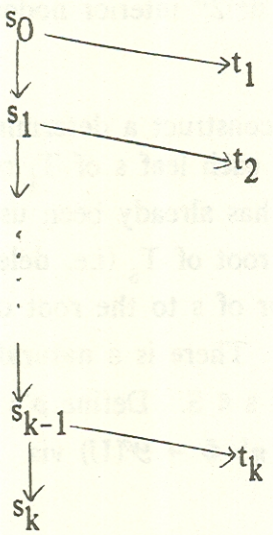
We need just one more technical lemma.

4.6 *Lemma*: For each $s \in S$ we can construct a tree T_s whose nodes are labelled by elements of S and whose edges are labelled by elements of $\Sigma_0(p_0)$ such that

(a) the root is labelled by s,
(b) if there is an edge labelled by A from s_1 to s_2, then $(s_1, s_2) \in \rho(A)$,
(c) for each node s′ on the tree and for each $A \in \Sigma_0(p_0)$, there is at most one edge labelled by A leading from s′ (i.e. the tree is deterministic),
(d) every formula of D(s) is fulfilled for s by some node on the tree,
(e) if s′ is any node on the tree and $q \in D(s')$, then either
  (i) q is fulfilled for s′ by t, where t is a descendant of s′ on the tree, or
  (ii) there is a leaf t′ on the tree which is a descendant of s′, and a derivative of q for s′ at t′.

Proof: For ease of exposition we will assume $\Sigma_0(p_0) = \{A_1, A_2\}$. Given $s_0 \in S$ with $D(s_0) = \{q_1, \ldots, q_m\}$. Suppose $q_1 = \langle A\rangle\langle a_1\rangle\ldots\langle a_h\rangle p$, where p is not of the form $\langle b\rangle r$ (of course A will be either A_1 or A_2). By Lemma 4.5, there is an $A;a_1;\ldots;a_h$ trajectory in M of length $\leq n\cdot 2^n$, say $(s_0, \ldots, s_k)$, such that $M, s_k \models p$. Note that each of $\langle a_1\rangle\ldots\langle a_h\rangle p$, $\ldots$, $\langle a_h\rangle p$, and p is also satisfied somewhere along this trajectory. Construct the straight line graph with nodes labelled by $s_0, s_1, \ldots, s_k$.

For all $i < k$ label the edge from s_i to s_{i+1} with A_j ($j = 1$ or $j = 2$) iff $(s_i, s_{i+1}) \in \rho(A_j)$. If s_{i+1} is an A_1 successor of s_i and not an A_2 successor of s_i, and if s_i has A_2 successors in M, add *one* of the A_2 successors of s_i to the graph, say t_{i+1}, and label the edge from s_i to t_{i+1} with A_2. Similarly if s_{i+1} is an A_2 successor and not an A_1 successor of s_i. So, for $i < k$, s_i has an A_j successor on the tree iff s_i has an A_j successor in M ($j = 1$ or $j = 2$). This gives us the following rather "thorny" tree, which we call *the thorny tree rooted at s fulfilling* q_1:



All edges are labelled by either A_1, A_2, or both.

So far we have a tree satisfying (a), (b), and (c) in which q_1 is fulfilled. We claim in addition that condition (e) is satisfied. It is trivially satisfied at s_k since s_k is a leaf. We show by induction on i that it is also satisfied at s_{k-i}. For suppose $q \in D(s_{k-i})$ and q is of the form $\langle A_j \rangle p$. Then either M, $s_{k-i+1} \models p$ or M, $t_{k-i+1} \models p$, depending on which one is the A_j successor of s_{k-i}. (This is precisely where we need the fact that M is a partial D model. The t's were chosen arbitrarily, but the D model condition ensures that $t_{k-i+1} \models p$, no matter what t is chosen as t_{k-i+1}).) Suppose M, $s_{k-i+1} \models p$. Then p is either immediately fulfilled at s_{k-i+1} or some $q' \in D(s_{k-i+1})$ is a derivative of p for s_{k-i+1} at s_{k-i+1}, and hence a derivative of q for s_{k-i} at s_{k-i+1}. By the inductive assumption, (e) holds for q' and hence also for q by the comments at the end of 4.4. If M, $t_{k-i+1} \models p$ the same argument holds without the appeal to the induction assumption, since t_{k-i+1} is already a leaf on the tree. Essentially, derivatives of q keep percolating their way down the tree until either one gets fulfilled or reaches a leaf of the tree.

We must still arrange to satisfy condition (d). Suppose q_2 is not fulfilled on the tree thus far constructed. Then by the argument above there is a leaf t on the tree and $q' \in D(t)$ which is a derivative of q_2 for s_0 at t. Now we just repeat the above construction. We append a thorny tree rooted at t which fulfills q'. It is easy to check that conditions (a), (b), (c), and (e) are still satisfied, and since q' is fulfilled for t, q_2 is fulfilled for s_0. We continue appending thorny trees in this way until all of $q_1, \ldots, q_m$ are fulfilled.

Note that since $m \leq n$, and each thorny tree which is appended has $\leq n \cdot 2^n$ interior nodes, the resultant tree has $\leq n^2 \cdot 2^n$ interior nodes. ∎

Proof of Theorem 4.1 (continued): We construct a deterministic tree T in stages. Let T_0 be s_0. Let T_{i+1} be T_i with each leaf s of T_i replaced by the tree T_s constructed above, *unless* T_s has already been used previously. In this case, identify s with the root of T_s (i.e. delete the leaf s and draw an edge from the predecessor of s to the root of T_s). Then let $T = \cup_i T_i$. Let U be the set of nodes on T. There is a natural map $\sigma: U \to S$ such that $\sigma(u) = s$ if $u \in U$ is an instance of $s \in S$. Define $\rho'': \Sigma_0 \to \mathcal{P}(U \times U)$ via $\rho''(A) = \{(u, u') | (\sigma(u), \sigma(u')) \in \rho(A)\}$ and $\pi': \Phi \to \mathcal{P}(U)$ via $\pi'(p) = \{u | M, \sigma(u) \models p\}$.

We extend ρ'' to $\rho': \Sigma \to \mathcal{P}(U \times U)$ in the usual way. It is not hard to see that (U, π', ρ') is a partial DPDL model for p_0. The only condition that must be checked is 7'. From Lemma 4.6(b), it follows that $(u, u') \in \rho(a)$ implies $(\sigma(u), \sigma(u')) \in \rho(a)$. Thus, if $U, u' \models p$ and $(u, u') \in \rho'(a)$, then $U, u \models \langle a \rangle p$ since $M, \sigma(u) \models \langle a \rangle p$.

For the converse, suppose $U, u \models \langle a \rangle p$. If $\langle a \rangle p$ is not immediately fulfilled by u, then by Lemma 4.3 there is a derivative of $\langle a \rangle p$, say q, in D(s). To show that there is some u' such that $U, u' \models p$ and $(u, u') \in \rho'(a)$, it suffices to show that q is fulfilled for u. But if u was first added to T when T_i was constructed, then by Lemma 4.6(e) q is fulfilled for u by some node in T_i or T_{i+1}.

Finally, note that there are at most 2^n distinct trees T_s (since $|S| \leq 2^n$), and each one has at most $n \cdot 2^n$ interior (non-leaf) nodes. Thus $|U| \leq n^2 \cdot 4^n$ (since leaves on one tree are always identified with interior nodes of some other tree in the construction), giving us the desired bound on the size of the partial model. ∎

5. Complexity

Theorem 4.1 can be applied to give a fast procedure for deciding whether a formula p_0 is DPDL satisfiable. An algorithm which takes nondeterministic time 2^{cn} for some constant c is almost immediate. Namely, we guess a partial D model $M = (S, \pi, \rho)$ of size $\leq 2^n$ and some $s \in S$, and test if $p_0 \in s$. If so, answer yes. But we can do better than this, by suitably modifying an algorithm of Pratt ([7]) for deciding PDL satisfiability:

5.1 *Theorem*: There is a procedure for deciding whether a formula p_0 is DPDL satisfiable which runs in deterministic time 2^{cn} for some constant c.

Proof: Let S_0 be the set of subsets of $FL(p_0)$.

1. For each $s \in S_0$, check that each of the following conditions hold:

   (a) if $\neg p \in FL(p_0)$, $\neg p \in s \leftrightarrow p \notin s$
   (b) if $\langle a \cup b \rangle p \in FL(p_0)$, $\langle a \cup b \rangle p \in s \leftrightarrow \langle a \rangle p \in s$ or $\langle b \rangle p \in s$
   (c) if $\langle a;b \rangle p \in FL(p_0)$, $\langle a;b \rangle p \in s \leftrightarrow \langle a \rangle \langle b \rangle p \in s$
   (d) if $\langle a^* \rangle p \in FL(p_0)$, $\langle a^* \rangle p \in s \leftrightarrow \langle a \rangle \langle a^* \rangle p \in s$ or $p \in s$
   (e) if $\langle p? \rangle q \in FL(p_0)$, $\langle p? \rangle q \in s \leftrightarrow p, q \in s$

   If any of the above conditions do not hold, eliminate s from S_0. Let S_1 be the remaining sets.

2. Consider the elements of S_1 as nodes on a graph. For each $A \in \Sigma_0(p_0)$, $s, t \in S_1$, join s to t by an edge labelled A *unless*

   (a) $\langle A \rangle p \in s$ and $p \notin t$ or
   (b) $\langle A \rangle p \in FL(p_0)$, $\langle A \rangle p \notin s$ and $p \in t$

3. Define ρ on $\Sigma(p_0)$ so that $\rho(A) = \{(s, t) |$ there is an edge from s to t labelled A$\}$. Compute $\rho(a)$ in the usual way for each program a that appears in p_0. Then for each node s on the graph, if $\langle a \rangle p \in FL(p_0)$, check that $\langle a \rangle p \in s$ implies that for some t with $(s, t) \in \rho(a)$ and $p \in t$. Eliminate s and all edges leading to and from s if it does not satisfy this condition, and repeat step 3 until all remaining nodes s do satisfy the condition.

Step 3 will be repeated at most $|S_1| \leq 2^n$ times. As well, as noted by Pratt ([7]), the computation of $\rho(a)$ and the necessary checking can be

carried out in time polynomial in the number of nodes remaining in the graph, again $\leq 2^n$.

4. Let S_2 be the remaining subsets. Then p_0 is satisfiable iff for some $s \in S_2$, $p_0 \in s$.

The comments made in step 3 justify the claim that the algorithm runs in deterministic time $O(c^n)$. To see that the algorithm is correct, first suppose that $p_0 \in s$ for some $s \in S_2$. Then we claim that $M = (S_2, \pi, \rho)$ is a partial D model for p_0, where π is defined so that $s \in \pi(p)$ iff $p \in s$. Step 2 in the algorithm guarantees that for $\langle A\rangle p \in FL(p_0)$, if $\langle A\rangle p \in s$ then $\forall t((s, t) \in \rho(A) \rightarrow p \in t)$. Otherwise, if for *any* $\langle A\rangle p \in s$ we have $p \notin t$, then (a) would have prevented the addition of (s, t) to $\rho(A)$. Step 3 implies that for $\langle a\rangle p \in FL(p_0)$, $\langle a\rangle p \in s \rightarrow \exists t((s, t) \in \rho(a) \wedge p \in t)$. It remains to show that if $\langle a\rangle p \in FL(p_0)$, $p \in t$, and $(s,t) \in \rho(a)$, then $\langle a\rangle p \in s$. This can be shown by induction on the structure of a. Step 2 guarantees that the statement is true for $\langle A\rangle p$. (The proviso $\langle A\rangle p \in FL(p_0)$ is used in the case, say, that $p_1, p_2 \in t$, $\langle A\rangle p_1 \in s$, but $\langle A\rangle p_2 \notin FL(p_0)$ and thus not in any state of S_1. That should not prevent adding (s, t) to $\rho(A)$ to fulfill $\langle A\rangle p$.) Using the conditions checked in step 1 we can show that the statement remains true for $\langle a \cup b\rangle p$, $\langle a;b\rangle p$, and $\langle p?\rangle q$. Now suppose $(s, t) \in \rho(a^*)$ and $p \in t$. Let $(s_0, \ldots, s_k)$ be an a^*-trajectory from s to t. Then we can show by induction on i that $\langle a^*\rangle p \in s_{k-i}$ for $0 \leq i \leq k$, using the main induction hypothesis and the condition (checked in step 1) that $\langle a^*\rangle p \in s$ iff $\langle a\rangle\langle a^*\rangle p \in s$ or $p \in s$. Finally, since there is a partial D model for p_0 of size $\leq 2^n$, by Theorem 4.1 p_0 is DPDL satisfiable.

For the converse, suppose p_0 is DPDL satisfiable. Then by Theorem 4.1, there is a partial D model for p_0 of size $\leq 2^n$, say $M' = (S', \pi', \rho')$. Let $f: S' \rightarrow S_1$ via $f(s') = \{p \in FL(p_0) \mid s' \in \pi'(p)\}$. Since for some $s' \in S'$, $s' \in \pi'(p_0)$, we must have that for some $s' \in S'$, $p_0 \in f(s')$. Then it is easily checked that after labelling the edges in step 2, we have for all $s_1, s_2 \in S'$

$$(s_1, s_2) \in \rho'(A) \rightarrow (f(s_1), f(s_2)) \in \rho(A)$$

It then follows that if $s' \in S'$, $f(s')$ will not be eliminated at step 3. Hence for some $s \in S_2$, $p_0 \in s$. ∎

5.2 *Remarks*: 1. If p_0 is DPDL satisfiable, the constructions in Theorems 4.1 and 5.1 actually give us an effective method for constructing a partial DPDL model for p_0 in time $2^{c'n}$ for some constant c'.

2. Parikh has shown ([4]) that the problem of deciding if a formula is PDL satisfiable is at least as hard as that of deciding if a formula is DPDL satisfiable. And by results of Fischer and Ladner ([1]), we know that there is some constant $d > 1$ such that no procedure can decide if an arbitrary formula of length n is PDL satisfiable in deterministic time $< 2^{dn}$. (Actually, Fischer and Ladner only seem to show that the formula cannot be decided in deterministic time $< 2^{dn/\log n}$. But here they are measuring the length of the formula in bits rather than in terms of the symbols of $\mathcal{L}$. If, as we have been doing in this paper, we measure the length of the formula in terms of symbols of $\mathcal{L}$, we get the 2^{dn} lower bound). Putting these two results together with Theorem 5.1, we see that we have tight bounds on the decision procedure for DPDL satisfiability.

5.3 *Algorithm*: The algorithm presented in 5.1 has, as noted in 5.2, the best possible worst case running time of 2^{cn}. However, its average case performance must also be 2^{cn}, since the first step involves creating all the subsets of $FL(p_0)$. We now sketch an algorithm that seems likely to do much better in most cases, since it uses a "bottom-up" approach, constructing only as much of the partial D model for p_0 as it needs.

We create a tree-like structure with nodes labelled by formulas in $FL(p_0) \cup \neg FL(p_0)$ (where $\neg FL(p_0) = \{\neg q \mid q \in FL(p_0)\}$).

There are two kinds of formulas:

1. α-formulas are those of the form $\langle p?\rangle q$ / $\langle a;b\rangle q$ / $\neg\langle a;b\rangle q$ / $\neg\langle a \cup b\rangle q$ / $\neg\langle a^*\rangle q$. They have successor sets $\{p, q\}$ / $\{\langle a\rangle\langle b\rangle q\}$ / $\{\neg\langle a\rangle\langle b\rangle q\}$ / $\{\neg\langle a\rangle q, \neg\langle b\rangle q\}$ / $\{\neg q, \neg\langle a\rangle\langle a^*\rangle q\}$ respectively.

2. β-formulas are those of the form $\langle a \cup b\rangle q$ / $\langle a^*\rangle q$ / $\neg\langle p?\rangle q$. They have successor sets $\{\langle a\rangle q, \langle b\rangle q\}$ / $\{\langle a\rangle\langle a^*\rangle q, q\}$ / $\{\neg p, \neg q\}$ respectively.

We build the tree inductively, level by level. The root is labelled by $\{p_0\}$. Suppose a node is labelled by Γ. We add successors to this node and label them by using the following three rules:

1. α-rule: if an α-formula is an element of Γ, add a successor node labelled by $\Gamma \cup$ (the successor set of that formula).

2. β-rule: if a β-formula is an element of Γ, add two successor nodes labelled by $\Gamma \cup \{$the i^{th} successor of the formula$\}$ $(i = 1, 2)$.

3. γ-rule: described below.

We apply the α- and β-rules to the leaves of the tree until all new nodes that would be obtained are already leaves on the tree. At this point we eliminate from further consideration all leaves labelled by sets Γ which do not satisfy all the following criteria:

(a) $\langle a;b\rangle p \in \Gamma \leftrightarrow \langle a\rangle\langle b\rangle p \in \Gamma$,
(b) $\langle a \cup b\rangle p \in \Gamma \leftrightarrow \langle a\rangle p \in \Gamma$ or $\langle b\rangle p \in \Gamma$,
(c) $\langle a^*\rangle p \in \Gamma \leftrightarrow p \in \Gamma$ or $\langle a\rangle\langle a^*\rangle p \in \Gamma$,
(d) $\langle p?\rangle q \in \Gamma \leftrightarrow p, q \in \Gamma$,
(e) $\langle a\rangle p \in \Gamma \leftrightarrow$ some *eventual successor* of $\langle a\rangle p$ of the form $\langle A\rangle\langle b_1\rangle \ldots \langle b_m\rangle p \in \Gamma$ or $p \in \Gamma$. (The eventual successors of $\langle a\rangle p$ are the elements of the least set containing $\langle a\rangle p$ and closed with respect to α and β successors. Note that we can keep track of the eventual successors of $\langle a\rangle p$ as we go along, so this condition is easy to check.)

At this point we apply the γ-rule to the remaining leaves: if Γ is the label of some leaf still under consideration and there is some formula of the form $\langle A\rangle p \in \Gamma$, create an A-successor of this leaf labelled with $\{p \mid \langle A\rangle p \in \Gamma\} \cup \{\neg q \mid \neg\langle A\rangle q \in \Gamma\}$. However, just as in the construction in 4.1, if we are about to apply the γ-rule to a node labelled by Γ and the γ-rule has already been applied to a node labelled by Γ, then we identify these two nodes.

Call the tree thus obtained T. Let $U = \{s \mid$ s is a node of T still under consideration before an application of the γ-rule$\}$. Define $\rho : \Sigma_0 \leftrightarrow U \times U$ via

$(s, t) \in \rho(A)$ iff t is a descendant, by application of α- and β-rules only, of an A-successor of s.

As in step 3 of the algorithm in 5.1, compute $\rho(a)$ for each program which a that appears in p_0. Then for each node $s \in T$, check that for each formula $\langle a\rangle p \in \Gamma_s$ (where Γ_s is the label of node s) there is some node t with $p \in \Gamma_t$ (actually it can be shown that it is sufficient to check this only for the cases where a is a primitive program or of the form b^*.) Eliminate s and all edges leading to and from s if it does not satisfy this condition, and repeat this step until all remaining nodes do satisfy this condition. Then it can be shown, using ideas similar to those in the proof of Theorem 5.1, that p_0 is satisfiable iff for some node s which does not get eliminated we have $p_0 \in \Gamma_s$. We omit the details here.

6. A Complete Axiomatization for DPDL

6.1 Consider the following deductive system for DPDL:

Axiom Schemes:

1. All tautologies of propositional calculus.
2. $\langle a \cup b \rangle p \leftrightarrow \langle a \rangle p \vee \langle b \rangle p$.
3. $\langle a \rangle (p \vee q) \leftrightarrow \langle a \rangle p \vee \langle a \rangle q$.
4. $\langle a;b \rangle p \leftrightarrow \langle a \rangle \langle b \rangle p$.
5. $\langle a^* \rangle p \leftrightarrow p \vee \langle a \rangle \langle a^* \rangle p$.
6. $\langle a^* \rangle p \rightarrow p \vee \langle a^* \rangle (\neg p \vee \langle a \rangle p)$.
7. $\langle A \rangle p \rightarrow [A]p$, for $A \in \Sigma_0$.

Inference Rules:

8. $\dfrac{p,\ p \rightarrow q}{q}$ (modus ponens)

9. $\dfrac{p}{[a]p}$ (generalization)

Axioms schemes 1-6 and rules 8, 9 constitute the Segerberg axioms for PDL and are known to give a complete axiomatization for PDL (see [2] for the easiest proof). We show how to combine Theorem 4.1 with the ideas of the Kozen-Parikh proof for the completeness of the Segerberg axioms for PDL to show that 1-9 give a complete axiomatization for DPDL.

6.2 *Theorem*: Axiom schemes and rules 1-9 above give a complete axiomatization for DPDL. *Proof*: We say that a formula p is *provable*, and write $\vdash p$, if there exis a finite sequence of formulas, the last one being p, wuch that each formula is an instant of an axiom scheme or follows from previous formulas by one of the inference rules. A formula p is *consistent* if not $\vdash \neg p$, i.e. if $\neg p$ is not provable in this system. We want to show that any valid DPDL formula is provable. It suffices to show that if p_0 is consistent, then p_0 is DPDL satisfiable.

So suppose p_0 is consistent. Let $FL(p_0) = \{q_1, \ldots, q_k\}$ ($k \leq |p_0|$). If s is a subset of $FL(p_0)$, let p_s, the *atom associated with s*, be the formula

$$(\wedge_{q_i \in s} q_i) \wedge (\wedge_{q_i \notin s} \neg q_i)$$

Let $S = \{s \subseteq FL(p_0) \mid p_s \text{ is consistent}\}$

For $s, t \in S$, define $\rho' : \Sigma_0 \to \mathcal{P}(S \times S)$ via

$(s, t) \in \rho'(A)$ iff $p_s \wedge \langle A \rangle p_t$ is consistent.

Define $\pi' : \Phi_0 \to \mathcal{P}(S)$ via

$s \in \pi'(P)$ iff $\vdash p_s \to P$ (iff P is one of the conjuncts in p_s).

Extend ρ', π', in the usual way to $\rho : \Sigma \to \mathcal{P}(S \times S)$, $\pi : \Phi \to \mathcal{P}(S)$.

Consider the structure $M = (S, \pi, \rho)$. It can easily be shown (see [2]) that if $q \in FL(p_0)$ or $q = \neg r$ and $r \in FL(p_0)$ then

$M, s \models q$ iff $\vdash p_s \to q$.

Moreover, any $q \in FL(p_0)$ is propositionally equivalent to the disjunction of all p_s such that $\vdash p_s \to q$. Since p_0 is consistent, there must be some s such that $\vdash p_s \to p_0$ and hence $M, s \models p_0$.

Thus M is a partial PDL model for p_0 of size $\leq 2^n$. But in fact M is a partial D model for p_0. To see this, suppose not. Then for some formula $\langle A \rangle q \in FL(p_0)$, we have $s, t \in S$ such that

$(s, t) \in \rho(A)$, $M, s \models \langle A \rangle q$, and $M, t \models \neg q$.

It then follows that $p_s \wedge \langle A \rangle p_t$ is consistent, $\vdash p_s \to \langle A \rangle q$, and $\vdash p_t \to \neg q$. But this implies that $\langle A \rangle q \wedge \langle A \rangle \neg q$ is consistent, contradicting Axiom 7.

Finally, by Theorem 4.1, since M is a partial D model for p_0, p_0 is DPDL satisfiable. ∎

7. Conclusion

We have given a 2^{cn} decision procedure for satisfiability in deterministic propositional dynamic logic. In addition we have shown that DPDL has the finite model property. The proof is unusual in the following sense. Usually one proves the finite model property (and a bound on the size of the

model) and then claims the obvious decision procedure: check all "small" models. Clearly a finite model is a by-product of the decision procedure. In our proof the decision procedure does not build a model. If you are only interested in satisfiability, then Theorem 4.1 shows it is sufficient to build the partial D model.

Acknowledgments

We would like to thank Rohit Parikh and Jonathan Stavi for many stimulating discussions on DPDL, and Vaughan Pratt for pointing out an error in the original proof of Theorem 5.1. Jonathan Stavi independently noted that the finite model property could be proved separately from the decision procedure.

References

1. M. J. Fischer and R. E. Ladner, Propositional modal logic of programs, *in* "Proceedings of the Ninth Annual ACM Symposium on Theory of Computing", 286-294, Association for Computing Machinery, New York, N. Y., 1977. A revised version appears as: Propositional dynamic logic of regular programs, *Journal of Computer and System Science* 18 (1979), 194-211.

2. D. Kozen and R. Parikh, An elementary proof of the completeness of PDL, to appear in *Theoretical Computer Science.*

3. R. Parikh, "A decidability result for Second Order Process Logic", Technical Report MIT/LCS/TM-112, M.I.T., 1978.

4. R. Parikh, Propositional logics of programs: systems, models, and complexity, *in* "Seventh Annual ACM Symposium on Principles of Programming Languages", 186-192, 1980.

5. V. R. Pratt, Semantical considerations of Floyd-Hoare logic, *in* "17th IEEE Symposium on the Foundations of Computer Science", 109-121, 1976.

6. V. R. Pratt, A practical decision method for propositional dynamic logic, *in* "10th ACM Symposium on the Theory of Computation", 326-337, 1977. A revised version appears as: A near optimal method for reasoning about action, *Journal of Computer and Systems Science* 20 (1980), 231-254.

7. V. R. Pratt, Models of program logics, *in* "20th IEEE Symposium on the Foundations of Computer Science", 115-122, 1979.

8. K. Segerberg, A completeness theorem in the modal logic of programs, Preliminary report, *Notices of the American Mathematics Society* 24 (1977), A552.

9. M. K. Valiev, On axiomatization of deterministic propositional dynamic logic, *in* "Symposium on the Mathematical Foundations of Computer Science, 1979", 482-491.

10. M. K. Valiev, Decision complexity of variants of propositional dynamic logic, *in* "Symposium on the Mathematical Foundations of Computer Science, 1980", 656-664.