COMPLETENESS AND EXPRESSIVE POWER IN 
PROPOSITIONAL DYNAMIC LOGIC

by

Fiona Humphris
B.Sc.(Hons) Victoria University of Wellington, 1991

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
in the Department of Mathematics and Statistics
of
Simon Fraser University

© Fiona Humphris 1995
SIMON FRASER UNIVERSITY
July 1995

All rights reserved. This work may not be
reproduced in whole or in part, by photocopy
or other means, without the permission of the author.
APPROVAL

Name: Fiona Humphris
Degree: M.Sc.
Title of thesis: Completeness and Expressive Power in Propositional Dynamic Logic

Examining Committee: Dr. G. A. C. Graham
Chair

Dr. S. K. Thomason, Senior Supervisor

Dr. A. H. Lachlan

Dr. N. R. Reilly

Dr. A. Gupta, External Examiner
Simon Fraser University

Date Approved: July 18, 1995
I hereby grant to Simon Fraser University the right to lend my thesis, project or extended essay (the title of which is shown below) to users of the Simon Fraser University Library, and to make partial or single copies only for such users or in response to a request from the library of any other university, or other educational institution, on its own behalf or for one of its users. I further agree that permission for multiple copying of this work for scholarly purposes may be granted by me or the Dean of Graduate Studies. It is understood that copying or publication of this work for financial gain shall not be allowed without my written permission.

Title of Thesis/Project/Extended Essay

Completeness and Expressive Power in Propositional Dynamic Logic

Author: ____________________________

(signature)

Fiona Humphris

(name)

24 May 1995

(date)
Abstract

Dynamic logics are a family of logics used to reason about computer programs. The language used is multi-modal, with modalities indexed by a set of programs and satisfying laws corresponding to the structure of the programs. We consider Propositional Dynamic Logic (PDL) which allows for two sorts of non-determinism in its programs, Strict Propositional Dynamic Logic (SPDL) which has no non-determinism, and two extensions of SPDL which each allow one form of non-determinism. We present soundness and completeness results for each of the logics, and compare their expressive powers. Wilm showed that the extension of SPDL by $\cup$ (SPDL+$\cup$) has less expressive power than PDL by showing there are conditions satisfied by every formula and program of SPDL+$\cup$ which are not satisfied by some formulas of PDL. We give an exposition of this result using simpler conditions than those used by Wilm. We also give results showing that the extension of SPDL by $*$ (SPDL+$*$) has expressive power equivalent to that of PDL, and SPDL has less expressive power than each of the other logics.
Acknowledgements

I would like to thank all the people who helped make writing this thesis possible, especially Steve Thomason and Alistair Lachlan.
Contents

Abstract iii

Acknowledgements iv

1 Introduction 1

2 Definitions 3
  2.1 Preliminaries 3
  2.2 Syntax of PDL and SPDL 4
  2.3 Semantics for PDL and SPDL 7
  2.4 Extensions of SPDL 9

3 Soundness and Completeness 11
  3.1 Preliminaries 11
  3.2 Soundness of PDL 12
  3.3 Completeness of PDL 14
  3.4 Soundness of SPDL 24
  3.5 Completeness of SPDL 26
  3.6 Soundness and Completeness of SPDL+U and SPDL+* 34

4 Expressive Power 35
  4.1 Definitions 35
  4.2 Expressive Power of PDL and SPDL 36
  4.3 Extensions of SPDL 39

Bibliography 55
Chapter 1

Introduction

Dynamic logics are a family of logics used to reason about computer programs. The concept was first introduced by Pratt [9] in 1976, who described first order dynamic logic. Here we are concerned with Propositional Dynamic Logic (PDL), which was introduced by Fisher and Ladner [1] in 1977, and a variant of PDL called Strict Propositional Dynamic Logic (SPDL). Both PDL and SPDL consist of a set of formulas and a set of programs. These are specified by giving a set of atomic formulas (the propositional variables), a set of atomic programs, and rules for constructing well formed formulas and programs. We have two notions of halting for programs. A program is said to terminate if it ends normally and is said to abort if it ends abnormally, for example if it is interrupted. We do not assume that every execution of a program will end.

To form complex programs in PDL the operations used are concatenation, denoted by ";", disjunction, meaning “do α or do β”, this is denoted by “∪” and is non-deterministic, an operator “∗”, meaning repeat a program some finite number of times, which is also non-deterministic, and a test operator denoted “?”, where F? means continue if formula F is true and abort otherwise. Formulas are constructed using negation and disjunction, and corresponding to each program α there is a modal operator [α], where [α]F is read “the formula F holds after every terminating execution of the program α”. We define the dual operator ⟨α⟩ by ⟨α⟩ =_{def} ¬[α]¬. The formula ⟨α⟩F is read “there is some execution of α after which formula F is true”. We are able to express assertions about programs within the language. We write ⟨α⟩true for “α terminates”, and use formulas such as F → [α]G to
talk about program correctness, and $[\alpha]F \leftrightarrow [\beta]F$ to talk about program equivalence.

In reality, no computer language in use today has any non-deterministic operators. Hence we are interested in the logic SPDL, which has no non-determinism. In SPDL, complex programs are constructed using the operations concatenation, “while do” and “if then else”, and we have as constants “skip” meaning “do nothing”, and “abort” meaning “abort”. The programs of SPDL have a similar structure to those one can write in a programming language such as C. Formulas of SPDL are constructed as for PDL. We also consider two extensions of SPDL. The first is the extension of SPDL by the operator $U$ (SPDL+$U$), and the second is the extension of SPDL by $*$ (SPDL+$*$).

We define a class of standard models for each logic, and show that the logics are both sound and complete with respect to the appropriate class of models. These results are from Goldblatt [2], [3] and [4].

Equivalence of formulas and programs of the different logics can be defined, and this enables us to compare the expressive powers of the four logics.

It was shown by Halpern and Reif in [5] that PDL has greater expressive power than SPDL. This motivates us to consider the extensions of SPDL by each of the non-deterministic operators “$U$” and “$*$”. We show that both of these extensions have greater expressive power than SPDL. We are now interested in whether the expressive power of either of the extensions is as great as the expressive power of PDL. We show that the logic SPDL+$*$ has expressive power equivalent to PDL, and the logic SPDL+$U$ has less expressive power than PDL. We give an exposition of a proof by Wilm [10] that the extension of SPDL by $U$ has less expressive power than PDL. The main part of the result is to show there are conditions satisfied by every formula and program of SPDL+$U$ which are not satisfied by some formula of PDL. Wilm uses a uniform finiteness condition on an infinite class of models, we use a simpler finiteness condition for the same class of models. We also give details of the induction proof which were omitted by Wilm.

Hence adding the $U$ form of non-determinism to SPDL gives less expressive power than adding the $*$ form of non-determinism.
Chapter 2

Definitions

2.1 Preliminaries

Definition 2.1.1 For any set $S$ the identity function on $S$, denoted $id_S$, is defined as follows:

$$id_S = \{(s,s) : s \in S\}.$$

Definition 2.1.2 Let $R_\alpha$ and $R_\beta$ be binary relations over the set $S$. We define the binary relation $R_\alpha \circ R_\beta$ by

$$R_\alpha \circ R_\beta = \{(s,t) : (\exists u \in S)(s R_\alpha u \text{ and } u R_\beta t)\}.$$

Definition 2.1.3 Let $R$ be a binary relation over a set $S$. For every $n \geq 0$ define a binary relation $R^n$ over $S$ by

$$R^0 = id_S$$

$$R^{n+1} = R^n \circ R.$$

The ancestral relation $R^*$ of $R$ is the binary relation over $S$ defined by

$$R^* = \{(s,t) : (\exists n)(\exists s_0, s_1, \ldots s_n)(s = s_0, t = s_n, \text{ and } (s_{i-1}, s_i) \in R \text{ for } 1 \leq i \leq n)\}.$$

Clearly $s R^* t$ if and only if $s R^n t$ for some $n \geq 0$. 
2.2 Syntax of PDL and SPDL

The alphabet for the language of PDL consists of the following symbols:

(i) a set of propositional variables $\Phi = \{p, q, r \ldots\}$

(ii) a set of atomic programs $\Pi = \{\pi, \sigma, \tau \ldots\}$

(iii) the symbols $\neg, \lor, \land$, $true, false, ;, U, *, ?, [, ], (, and )$.

Let $W_{PDL}$ denote the set of words over this alphabet. For $A, B, U, V \in \mathcal{P}(W_{PDL})$ we define $(A, B) \subseteq (U, V)$ if $A \subseteq U$ and $B \subseteq V$. Let $(F_{PDL}, P_{PDL})$ denote the $\subseteq$-least pair $(F, P)$ in $\mathcal{P}(W_{PDL}) \times \mathcal{P}(W_{PDL})$ such that the following are satisfied:

F1: $\Phi \subseteq F$

F2: $true \in F$

F3: $false \in F$

F4: If $F \in F$ then $\neg F \in F$

F5: If $F, G \in F$ then $(F \lor G) \in F$

F6: If $F \in F$ and $\alpha \in P$ then $[\alpha]F \in F$.

P1: $\Pi \subseteq P$

P2: If $\alpha, \beta \in P$, then $(\alpha; \beta) \in P$

P3: If $\alpha, \beta \in P$, then $(\alpha \cup \beta) \in P$

P4: If $\alpha \in P$, then $\alpha^* \in P$

P5: If $F \in F$, then $F? \in P$.

$F_{PDL}$ is called the set of formulas of PDL and $P_{PDL}$ the set of programs of PDL.

The alphabet for the language of SPDL consists of the following symbols:
CHAPTER 2. DEFINITIONS

(i) a set of propositional variables $\Phi = \{p, q, r \ldots \}$

(ii) a set of atomic programs $\Pi = \{\pi, \sigma, \tau \ldots \}$

(iii) the symbols $\neg, \lor, \text{true}, \text{false}, ;, \text{skip}, \text{abort}, \text{while}, \text{do}, \text{if}, \text{then}, \text{else}, [, ], (, and )$.

Let $W_{\text{SPDL}}$ denote the set of words over this alphabet. Let $(F_{\text{SPDL}}, P_{\text{SPDL}})$ denote the $\subseteq$-least pair $(F, P)$ in $\mathcal{O}(W_{\text{SPDL}}) \times \mathcal{O}(W_{\text{SPDL}})$ which satisfies F1 - F6 and also the following:

$P1'$: $\Pi \subseteq P$

$P2'$: If $\alpha, \beta \in P$ then $(\alpha; \beta) \in P$

$P3'$: $\text{skip} \in P$

$P4'$: $\text{abort} \in P$

$P5'$: If $\alpha \in P$ and $F \in F$, then (while $F$ do $\alpha$) $\in P$

$P6'$: If $\alpha, \beta \in P$ and $F \in F$, then (if $F$ then $\alpha$ else $\beta$) $\in P$.

$F_{\text{SPDL}}$ is called the set of formulas of SPDL and $P_{\text{SPDL}}$ the set of programs of SPDL.

Below Greek letters $\alpha, \beta, \gamma \ldots$ denote programs, and upper case letters $E, F, G \ldots$ denote formulas.

The connectives are read as follows: ";" is concatenation so $\alpha;\beta$ means “do $\alpha$ then do $\beta$”, "\lor" is disjunction so $\alpha \lor \beta$ means “do $\alpha$ or $\beta$" (\lor is nondeterministic), $\alpha^*$ means “repeat $\alpha$ some finite number of times" (* is nondeterministic), $F?$ means “test $F$ and continue if it is true otherwise abort”, $[\alpha]F$ means “$F$ holds after every terminating execution of $\alpha$”.

The program $\text{skip}$ is “do nothing”, and $\text{abort}$ means “abort”.

We define $\land, \rightarrow$, and $\leftrightarrow$ as usual. The modal operator $\langle \rangle$, which is the dual of $[ ]$, is defined by $(\alpha)F \equiv_{df} \neg[\alpha]\neg F$ and is read “there is a terminating execution of $\alpha$ after which $F$ holds”.

Axiom schemas for PDL:

PC: All tautologies
CHAPTER 2. DEFINITIONS

K: \([\alpha](F \rightarrow G) \rightarrow ([\alpha]F \rightarrow [\alpha]G)\)

COMP: \([\alpha; \beta]F \leftrightarrow [\alpha][\beta]F\)

ALT: \([\alpha \cup \beta]F \leftrightarrow [\alpha]F \land [\beta]F\)

MIX: \([\alpha^*]F \rightarrow F \land [\alpha][\alpha^*]F\)

IND: \([\alpha^*](F \rightarrow [\alpha]F) \rightarrow (F \rightarrow [\alpha^*]F)\)

TEST: \([F?]G \leftrightarrow (F \rightarrow G)\).

Rules of Inference for PDL:

MODUS PONENS: From \(F\) and \((F \rightarrow G)\) infer \(G\)

NECESSITATION: From \(F\) infer \([\alpha]F\).

Let \(\text{Th}(\text{PDL})\) denote the least set \(T \subseteq \mathcal{F}_{\text{PDL}}\) such that \(T\) contains all instances of the axiom schemas and is closed under the rules of inference. The members of \(\text{Th}(\text{PDL})\) are called theorems of PDL.

Axiom schemas for SPDL:

PC: All tautologies

K: \([\alpha](F \rightarrow G) \rightarrow ([\alpha]F \rightarrow [\alpha]G)\)

DUM: \([\text{skip}]F \leftrightarrow F\)

ABORT: \([\text{abort}]\text{false}\)

COMP: \([\alpha; \beta]F \leftrightarrow [\alpha][\beta]F\)

COND: \([\text{if } E \text{ then } \alpha \text{ else } \beta]F \leftrightarrow ((E \rightarrow [\alpha]F) \land (\neg E \rightarrow [\beta]F))\)

WHILE1: \(\neg E \rightarrow ([\text{while } E \text{ do } \alpha]F \rightarrow F)\)

WHILE2: \([\text{while } E \text{ do } \alpha]F \rightarrow (E \rightarrow [\alpha][\text{while } E \text{ do } \alpha]F)\).
CHAPTER 2. DEFINITIONS

Rules of Inference for SPDL:

**Modus Ponens:** From $F$ and $(F \rightarrow G)$ infer $G$

**Necessitation:** From $F$ infer $[\alpha]F$

**Hoare’s Iteration Rule:** From $(E \land F) \rightarrow [\alpha]F$ infer $F \rightarrow [\text{while } E \text{ do } \alpha](\neg E \land F)$.

As above, let $\text{Th}(\text{SPDL})$ denote the least set $T \subseteq \mathcal{F}_{\text{SPDL}}$ such that $T$ contains all instances of the axiom schemas and is closed under the rules of inference. The members of $\text{Th}(\text{SPDL})$ are called *theorems* of SPDL.

### 2.3 Semantics for PDL and SPDL

In the following we use $\mathcal{P}$ to denote the set of programs of PDL or SPDL, whichever is appropriate. A *model* for either PDL or SPDL is a structure $\mathcal{M} = \langle S, \{R_\alpha : \alpha \in \mathcal{P}\}, V \rangle$, where $S$ is a nonempty set, $R_\alpha$ is a binary relation on $S$ for each $\alpha \in \mathcal{P}$, and $V$ is a function from $\Phi$ to $\mathcal{P}(S)$. The members of $S$ are called *states*. Intuitively the pairs in $R_\alpha$ are the possible *transitions* of the program $\alpha$, and $V(p)$ is the set of states for which $p$ is true.

Satisfiability is defined as a 3-ary relation between model, a state, and a formula as follows:

1. $\mathcal{M}, s \vdash p$ if $s \in V(p)$, for $p \in \Phi$
2. $\mathcal{M}, s \vdash \text{true}$
3. $\mathcal{M}, s \not\vdash \text{false}$
4. $\mathcal{M}, s \vdash \neg F$ if $\mathcal{M}, s \not\vdash F$
5. $\mathcal{M}, s \vdash F \lor G$ if $\mathcal{M}, s \vdash F$ or $\mathcal{M}, s \vdash G$
6. $\mathcal{M}, s \vdash [\alpha]F$ if for all $s' \in S$, $sR_\alpha s'$ implies $\mathcal{M}, s' \vdash F$. 
From the definitions of the symbols $\land, \rightarrow, \leftrightarrow$ and of the modal operator $\langle \rangle$ we obtain the following equivalences:

(i) $\mathcal{M}, s \models F \land G$ if and only if $\mathcal{M}, s \models F$ and $\mathcal{M}, s \models G$

(ii) $\mathcal{M}, s \models F \rightarrow G$ if and only if $\mathcal{M}, s \not\models F$ or $\mathcal{M}, s \models G$

(iii) $\mathcal{M}, s \models F \leftrightarrow G$ if and only if both $\mathcal{M}, s \models F$ and $\mathcal{M}, s \models G$, or both $\mathcal{M}, s \not\models F$ and $\mathcal{M}, s \not\models G$

(iv) $\mathcal{M}, s \models \langle \alpha \rangle F$ if and only if there is an $s' \in S$ such that $s R_\alpha s'$ and $\mathcal{M}, s' \models F$.

An equivalent definition of satisfaction is obtained by letting $\overline{V} : F \rightarrow \mathcal{P}(S)$ be the unique map satisfying the following conditions:

S1': $V \subseteq \overline{V}$

S2': $\overline{V}(true) = S$.

S3': $\overline{V}(false) = \emptyset$.

S4': $\overline{V}(\neg F) = S \setminus \overline{V}(F)$.

S5': $\overline{V}(F \lor G) = \overline{V}(F) \cup \overline{V}(G)$.

S6': $\overline{V}(\langle \alpha \rangle F) = \{s \in S : (\forall s' \in S)(s R_\alpha s' \Rightarrow s' \in \overline{V}(F))\}$.

Then satisfiability is defined by specifying that $\mathcal{M}, s \models F$ if $s \in \overline{V}(F)$.

Again from the definitions of the symbols $\land, \rightarrow, \leftrightarrow$, and the modal operator $\langle \rangle$ we obtain the following equivalences:

$\overline{V}(F \land G) = \overline{V}(F) \cap \overline{V}(G)$

$\overline{V}(F \rightarrow G) = (S \setminus \overline{V}(F)) \cup \overline{V}(G)$

$\overline{V}(F \leftrightarrow G) = (\overline{V}(F) \cap \overline{V}(G)) \cup ((S \setminus \overline{V}(F)) \cap (S \setminus \overline{V}(G)))$
A formula $F$ is valid in the model $\mathcal{M}$ if $\mathcal{M}, s \models F$ for every $s \in S$.

A standard model for PDL is a model in which the following conditions hold:

1. $R_{\alpha \beta} = R_{\alpha} \circ R_{\beta}$
2. $R_{\alpha \cup \beta} = R_{\alpha} \cup R_{\beta}$
3. $R_{\alpha^*} = (R_{\alpha})^*$
4. $R_{E \gamma} = \{(s, s): \mathcal{M}, s \models E\}$.

M1–M4 are called the standard model conditions for PDL.

A standard model for SPDL is a model in which M1 holds, and the following conditions also hold:

1. $R_{\text{skip}} = id_S$
2. $R_{\text{abort}} = \emptyset$
3. $R_{\text{while } E \text{ do } \alpha} = \{(s, s') : (\exists n)(\exists s_0, \ldots, s_n)(s = s_0, s' = s_n, \mathcal{M}, s_n \models \neg E \text{ and } (\forall j < n)((s_j, s_{j+1}) \in R_{\alpha} \text{ and } \mathcal{M}, s_j \models E))\}$
4. $R_{\text{if } E \text{ then } \alpha \text{ else } \beta} = \{(s, s') : sR_{\alpha}s' \text{ and } \mathcal{M}, s \models E\} \cup \{(s, s') : sR_{\beta}s' \text{ and } \mathcal{M}, s \models \neg E\}$.

M1 and M2'–M5' are the standard model conditions for SPDL.

A formula $F$ of PDL is valid, denoted $\models F$, if $F$ is valid in every standard model of PDL. Similarly a formula $G$ of SPDL is valid, denoted $\models G$, if $G$ is valid in every standard model for SPDL.
2.4 Extensions of SPDL

Let SPDL+\( \cup \) denote the extension of SPDL by \( \cup \), which is defined by adding the following to SDPL:

- \( \cup \) to the language of SPDL
- P3 to the program formation rules
- ALT to the axiom schemas
- M2 to the standard model conditions.

Let SPDL+\( * \) denote the extension of SPDL by \( * \), which is defined by adding the following to SDPL:

- \( * \) to the language of SPDL
- P4 to the program formation rules
- IND and MIX to the axiom schemas
- M3 to the standard model conditions.
Chapter 3

Soundness and Completeness

3.1 Preliminaries

Let \( \mathcal{L} \) be a logic such as PDL or SPDL, and \( M \) be a class of models for \( \mathcal{L} \). Let \( \text{Th}(\mathcal{L}) \) denote the class of theorems of \( \mathcal{L} \). Then \( \mathcal{L} \) is sound with respect to \( M \) if every theorem of \( \mathcal{L} \) is valid in every model in \( M \). \( \mathcal{L} \) is complete with respect to \( M \) if every formula which is valid in every model in \( M \) is a theorem of \( \mathcal{L} \).

**Definition 3.1.1** Let \( \Lambda \) be a set of formulas of a logic \( \mathcal{L} \). \( \Lambda \) is consistent if \( \neg(F_1 \land \ldots \land F_n) \) is a non-theorem of \( \mathcal{L} \) for every finite subset \( \{F_1, \ldots, F_n\} \subseteq \Lambda \).

**Definition 3.1.2** We define programs \( \alpha^n \) for each \( n \in \mathbb{N} \) as follows:

\[
\alpha^0 = \text{skip} \\
\alpha^{n+1} = \alpha^n ; \alpha.
\]

Note that in standard models \( \mathcal{M}, s \models [\alpha^*]F \) if and only if for every \( n \geq 0 \), \( \mathcal{M}, s \models [\alpha^n]F \), and that \( R_{\alpha^n} = (R_\alpha)^n \).

**Definition 3.1.3** The set of subformulas of a formula \( E \) of PDL is defined inductively as follows for every \( p \in \Phi, F, G \in \mathcal{F} \) and \( \alpha \in \mathcal{P} \):

\[
\]
CHAPTER 3. SOUNDNESS AND COMPLETENESS

Note that formulas occurring in $\alpha$ are not subformulas of $[\alpha]F$.

We show that PDL is sound and complete with respect to the class of standard models for PDL. The proof for soundness is given below; the completeness proof appears in the next section.

3.2 Soundness of PDL

**Theorem 3.2.1** PDL is sound with respect to the class of standard PDL models.

**Proof:** We need to show that each axiom of PDL is valid, and that the inference rules preserve validity. Below we only consider the last five axioms. The remaining axioms and transformation rules are contained in every normal modal logic, and the required proofs are straightforward. See for example [4] p25 or [7] pp 68–69. We treat each axiom schema in turn. Below $\mathcal{M}$ denotes a standard model for PDL.

**COMP:**

\[
\begin{align*}
\mathcal{M}, s \models [\alpha; \beta]F & \quad \overset{S_6}{\Rightarrow} \quad (\forall t)(sR_{\alpha}t \Rightarrow \mathcal{M}, t \models F) \\
\quad \overset{M_1}{\Rightarrow} \quad (\forall t)(sR_\alpha \circ R_\beta t \Rightarrow \mathcal{M}, t \models F) \\
\quad \overset{\equiv}{\Rightarrow} \quad (\forall t)(\forall u)(sR_\alpha u R_\beta t \Rightarrow \mathcal{M}, t \models F) \\
\quad \overset{\equiv}{\Leftarrow} \quad (\forall u)(sR_\alpha u \Rightarrow ((\forall t)(uR_\beta t \Rightarrow \mathcal{M}, t \models F))) \\
\quad \overset{S_6}{\Rightarrow} \quad (\forall u)(sR_\alpha u \Rightarrow \mathcal{M}, u \models [\beta]F) \\
\quad \overset{S_6}{\Rightarrow} \quad \mathcal{M}, s \models [\alpha][\beta]F.
\end{align*}
\]

Hence $\mathcal{M}, s \models [\alpha; \beta]F \Leftarrow ([\alpha][\beta]F)$. 


ALT:

\[ M, s \models [\alpha \cup \beta]F \quad \implies \forall t (sR_{\alpha \cup \beta}t \Rightarrow M, t \models F) \]

\[ \implies \forall t (sR_\alpha t \Rightarrow M, t \models F) \text{ and } \forall t(sR_\beta t \Rightarrow M, t \models F) \]

\[ \iff M, s \models [\alpha]F \text{ and } M, s \models [\beta]F \]

\[ \iff M, s \models [\alpha]F \land [\beta]F. \]

Hence \( M, s \models [\alpha \cup \beta]F \iff ([\alpha]F \land [\beta]F) \).

MIX: Suppose \( M, s \models [\alpha^*]F \), and let \( t \) be such that \( sR_\alpha t \). Then

\[ tR_\alpha u \quad \implies \quad t(R_\alpha)^*u \implies s(R_\alpha)^*u \implies sR_\alpha u \implies M, u \models F. \]

Thus for all \( t \), if \( sR_\alpha t \) then \( M, t \models [\alpha^*]F \), so \( M, s \models [\alpha][\alpha^*]F \). Also by definition \( sR_\alpha s \), hence \( M, s \models F \). Thus \( M, s \models [\alpha^*]F \implies F \). Hence \( M, s \models [\alpha^*]F \). Also by definition \( sR_\alpha s \), hence \( M, s \models F \). Thus \( M, s \models [\alpha^*]F \implies F \). Hence \( M, s \models [\alpha^*]F \) as required.

IND: Suppose \( M, s \models [\alpha^*](F \rightarrow [\alpha]F) \), and \( M, s \models F \). We need to show that \( M, s \models [\alpha^*]F \). We show by induction that \( M, s \models [\alpha^n]F \) for every \( n \geq 0 \). Case \( n = 0 \) is given, as \( M, s \models F \). For the induction step assume \( M, s \models [\alpha^k]F \). We want to show that \( M, s \models [\alpha^{k+1}]F \). Suppose \( sR_\alpha^{k+1}t \). Then there exists \( u \) such that \( sR_\alpha u \) and \( sR_\alpha u \). By the induction hypothesis \( sR_\alpha u \) yields \( M, u \models F \). Hence \( M, u \models [\alpha^k]F \). Now \( uR_\alpha t \) yields \( M, t \models F \). Hence \( sR_\alpha^{k+1}t \) implies \( M, t \models F \), so \( M, s \models [\alpha^{k+1}]F \) as required.

TEST:

\[ M, s \models [F?]G \quad \iff \quad \forall t(sR_? t \Rightarrow M, t \models G) \]

\[ \iff \forall t(s = t \text{ and } M, s \models F) \Rightarrow M, t \models G \]

\[ \iff M, s \models F \Rightarrow M, s \models G \]

\[ \iff M, s \models (F \rightarrow G). \]

Hence \( M, s \models [F?]G \iff (F \rightarrow G). \)

\( \Box \)
3.3 Completeness of PDL

In this section we will show that PDL is complete with respect to the class of standard PDL models. To do this we show that every non-theorem of PDL is not valid in some standard PDL model. We first construct the canonical model. The canonical model for PDL is the model $M^C = (S^C, \{R^C_\alpha : \alpha \in \mathcal{P}_{PDL}\}, V^C)$ where

- $S^C$ is the set of all maximal PDL-consistent sets of formulas
- for each $\alpha \in \mathcal{P}_{PDL}$, $R^C_\alpha$ is the binary relation defined by $s R^C_\alpha t \iff \{G \in \mathcal{F} : [\alpha]G \in s\} \subseteq t$
- $V^C : \Phi \rightarrow \mathcal{P}(S^C)$ is the function given by $V^C(p) = \{s \in S^C : p \in s\}$.

Lemma 3.3.1 Let $F$ be a formula of PDL. Then for every $s \in S^C$, $M^C, s \models F \iff F \in s$.


It follows that every theorem of PDL is valid in $M^C$, and no non-theorem of PDL is valid in $M^C$. However $M^C$ is not necessarily a standard model of PDL. We now construct for each non-theorem $A$ of PDL a standard model in which $A$ is not valid.

Definition 3.3.2 A set $\Gamma$ of PDL formulas is called FL-closed if it is closed under subformulas and the following conditions hold for all $F, G \in \mathcal{F}_{PDL}$ and $\alpha, \beta \in \mathcal{P}_{PDL}$:

(i) $[F?]G \in \Gamma \Rightarrow F \in \Gamma$

(ii) $[\alpha; \beta]F \in \Gamma \Rightarrow [\alpha][\beta]F \in \Gamma$

(iii) $[\alpha \cup \beta]F \in \Gamma \Rightarrow [\alpha]F \in \Gamma$ and $[\beta]F \in \Gamma$

(iv) $[\alpha^*]F \in \Gamma \Rightarrow [\alpha][\alpha^*]F \in \Gamma$.

Let $\Gamma$ be an FL-closed set of PDL formulas, and $M = (S, \{R_\alpha : \alpha \in \mathcal{P}\}, V)$ be a model. Define an equivalence relation on $S$ by

$s \sim t \iff (\forall F \in \Gamma)(M, s \models F \iff M, t \models F)$. 
CHAPTER 3. SOUNDNESS AND COMPLETENESS

We introduce the following notations:

- \([s] \) for \( \{ t \in S : s \sim t \} \)
- \( \Phi^\Gamma \) for \( \Phi \cap \Gamma \)
- \( \Pi^\Gamma \) for the set of atomic programs occurring in formulas in \( \Gamma \)
- \( \mathcal{P}^\Gamma \) for the closure of \( \Pi^\Gamma \cup \{ F? : F? \text{ occurs in a member of } \Gamma \} \) under the operations \( \cup, *, \) and \( ; \)
- \( S^\Gamma \) for \( \{ [s] : s \in S \} \).

Let \( V^\Gamma : \Phi^\Gamma \rightarrow \mathcal{P}(S^\Gamma) \) be defined by the equivalence

\[
[s] \in V^\Gamma(p) \iff s \in V(p).
\]

**Definition 3.3.3** For \( \alpha \in \mathcal{P}_{\text{PDL}} \), a binary relation \( R^\Gamma \) on \( S^\Gamma \) is a \((\Gamma, \alpha)\)-filtration of the binary relation \( R_\alpha \) on \( S \) if the following conditions hold for all \( s, t \in S \):

- **G1:** if \( sR_\alpha t \) then \([s]R^\Gamma[t]\)
- **G2:** for all \( F \in \mathcal{F} \), if \([s]R^\Gamma[t]\) and \([\alpha]F \in \Gamma \) and \( \mathcal{M}, s \models [\alpha]F \) then \( \mathcal{M}, t \models F \).

An example of a \((\Gamma, \alpha)\)-filtration of \( R_\alpha \) is the smallest filtration,

\[
R^\Gamma = \{ ([s], [t]) : (\exists s' \in [s])(\exists t' \in [t])(s'R_\alpha t') \}.
\]

The model \( \mathcal{M}^\Gamma = (S^\Gamma, \{ R^\Gamma_\alpha : \alpha \in \mathcal{P}^\Gamma \}, V^\Gamma) \) is a \( \Gamma \)-filtration of \( \mathcal{M} = (S, \{ R_\alpha : \alpha \in \mathcal{P} \}, V) \) if \( R^\Gamma_\alpha \) is a \((\Gamma, \alpha)\)-filtration of \( R_\alpha \) for every \( \alpha \in \mathcal{P}^\Gamma \).

**Lemma 3.3.4** Let \( \mathcal{M}^\Gamma = (S^\Gamma, \{ R^\Gamma_\alpha : \alpha \in \mathcal{P}^\Gamma \}, V^\Gamma) \) be a \( \Gamma \)-filtration of the model \( \mathcal{M} = (S, \{ R_\alpha : \alpha \in \mathcal{P} \}, V) \). Then for all \( F \in \Gamma \), \( \mathcal{M}, s \models F \) if and only if \( \mathcal{M}^\Gamma, [s] \models F \).

**Proof:** By induction on formulas. The details can be found in [8] p139 or [4] pp33, 115. \( \Box \)
Lemma 3.3.5 Suppose that $\mathcal{M}^\Gamma = (\mathcal{S}^\Gamma, \{R_\alpha^\Gamma : \alpha \in \mathcal{P}^\Gamma\}, V^\Gamma)$ is a $\Gamma$-filtration of the model $\mathcal{M} = (\mathcal{S}, \{R_\alpha : \alpha \in \mathcal{P}_{\text{PDL}}\}, V)$ such that $\Gamma$ is finite, and let $T$ be a subset of $\mathcal{S}^\Gamma$. Then there is a formula $\mathcal{A}_T$ such that for all $s \in \mathcal{S}$, $\mathcal{M}, s \models \mathcal{A}_T$ if and only if $[s] \in T$.

Proof: For each $t \in \mathcal{S}^\Gamma$ let $A_t$ be the conjunction of formulas in the set 
\{ $F : F \in \Gamma$ and $\mathcal{M}, t \models F$ \} $\cup \{ \neg F : F \in \Gamma$ and $\mathcal{M}, t \not\models F$ \}. This is a finite conjunction as $\Gamma$ is finite. Then

$$\mathcal{M}, s \models A_t \iff (\mathcal{M}, s \models F \iff \mathcal{M}, t \models F) \text{ for all } F \in \Gamma$$

$$\iff s \sim t$$

$$\iff [s] = [t].$$

As $|\mathcal{S}^\Gamma| \leq 2^{|\Gamma|}$, then $\mathcal{S}^\Gamma$ must also be finite, and hence $T$ is finite. If $T = \emptyset$ then set $\mathcal{A}_T = \text{false}$. Otherwise $T = \{[t_1], \ldots, [t_n]\}$. Let $\mathcal{A}_T = A_{t_1} \lor \ldots \lor A_{t_n}$. Then

$$\mathcal{M}, s \models \mathcal{A}_T \iff \mathcal{M}, s \models A_{t_1} \lor \ldots \lor \mathcal{M}, s \models A_{t_n}$$

$$\iff [s] = [t_1] \lor \ldots \lor [s] = [t_n]$$

$$\iff [s] \in T.$$
Lemma 3.3.6 $\Gamma_A$ is finite.

Proof: We define a formula to be modal if it has the form $[\alpha]G$. The modal length of $[\alpha]G$ is the number of symbol occurrences in $\alpha$. We show that in the construction of $\Gamma_A$, starting with $\{A\}$ and closing under conditions (i)-(iv) and subformulas, every formula can only cause a finite number of new formulas, of lesser modal length, to be added.

First observe the following for (ii)-(iv):


(iii) Every subformula of $[\alpha]F$ or $[\beta]F$, except for $[\alpha]F$ and $[\beta]F$, is a subformula of $[\alpha \cup \beta]F$,

(iv) Every subformula of $[\alpha][\alpha^*]F$, except for $[\alpha][\alpha^*]F$, is a subformula of $[\alpha^*]F$.

Hence if one of (ii)-(iv) is applied to a formula $[\alpha]G$ and we close under subformulas then only a finite number of formulas, each of which is modal with strictly smaller modal length than $[\alpha]G$, are added. If we apply (i) to a formula $[F?]G$ and close under subformulas then as $Sf(F)$ is finite and each subformula of $F$ is shorter than $F?$, then (i) also adds a finite number of formulas, where each new modal formula has strictly smaller modal length than $[F?]G$.

Note that the FL-closure of $\{A\}$ is the closure of $Sf(A)$ under (i)-(iv) and subformulas. Let $A_0 = Sf(A)$, and $A_{n+1}$ be the set of new formulas obtained by applying (i)-(iv) to formulas in $A_n$ and closing under subformulas. Then the greatest modal length for a formula in $A_{n+1}$ is strictly less than the greatest modal length for a formula in $A_n$. As every modal formula has modal length at least 1, it follows that there exists an $m$ such that $A_m$ contains no modal formulas. For $n > m$ we have $A_n = \emptyset$. We know that $Sf(A)$ is finite, and for each modal formula the conditions (i)-(iv) and closing under subformulas adds a finite number of new formulas, so $A_n$ is finite for every $n$. Hence $\Gamma_A = \bigcup_{n \leq m} A_n$ is finite. $\square$

Lemma 3.3.7 $\mathcal{M}^\Gamma$ is a $\Gamma_A$-filtration of $\mathcal{M}^C$. 

CHAPTER 3. SOUNDNESS AND COMPLETENESS

Proof: We need to show that $R^\Gamma_\alpha$ is a $(\Gamma_A, \alpha)$-filtration of $R^C_\alpha$ for $\alpha \in \mathcal{P}^\Gamma$. We use induction on the construction of programs.

Case 1: $\alpha \in \Pi^\Gamma$. Then $R^\Gamma_\alpha$ is a $\Gamma_A$-filtration of $R^C_\alpha$ by definition of $R^\Gamma_\alpha$.

Case 2: $\alpha = F?\!$ for some $F? \in \mathcal{P}^\Gamma$.

G1: We first show that $sR^C_{F?}t$ implies both $s = t$ and $\mathcal{M}^C, s \models F$. Suppose $sR^C_{F?}t$. Then we have

$$\{G \in \mathcal{F} : [F?]G \in s\} \subseteq t$$

from the definition of $R^C_{F?}$. To show that $s = t$ we note that

$$G \in s \implies F \rightarrow G \in s$$

$$\overset{\text{TEST}}{\iff} [F?]G \in s$$

$$\overset{(3.1)}{\implies} G \in t.$$ 

Hence $s \subseteq t$, but both $s$ and $t$ are maximal so we must have $s = t$. For the rest observe that

$$(F \rightarrow F) \in s \overset{\text{TEST}}{\iff} [F?]F \in s$$

$$\overset{(3.1)}{\iff} F \in t$$

$$\iff F \in s$$

$$\overset{3.3.1}{\iff} \mathcal{M}^C, s \models F.$$ 

Hence

$$sR^C_{F?}t \implies (s = t \text{ and } \mathcal{M}^C, s \models F)$$

$$\iff [s]R^\Gamma_{F?}[s].$$ 

so G1 holds.
G2: Suppose \([s]R_{F,G}^\Gamma t\), and also that \([F?]G \in \Gamma_A\) and \(\mathcal{M}^C, s \models [F?]G\). Then by definition of \(R_{F,G}^\Gamma\) we have \([t] = [s]\) and \(\mathcal{M}^C, s \models F\). The axiom schema TEST yields \(\mathcal{M}^C, s \models (F \rightarrow G)\), and hence \(\mathcal{M}^C, s \models G\). But \(G \in \Gamma_A\) (as \(\Gamma_A\) is closed under subformulas) and \(s \sim t\) (as \([s] = [t]\)) so \(\mathcal{M}^C, t \models G\).

Case 3: \(\alpha = \beta; \gamma\)

G1: Fix \(s\) and let \(A_s\) be a formula such that for all \(t, A_s \in t\) if and only if \([s]R_{\beta;\gamma}^\Gamma t\) (such an \(A_s\) exists by Lemma 3.3.5). By the induction hypothesis G1 holds for \(R_{\beta}^\Gamma\) and \(R_{\gamma}^\Gamma\). We now show that \(\mathcal{M}^C, s \models [\beta][\gamma]A_s\):

\[
\begin{align*}
\text{s}R_{\beta}^\Gamma uR_{\gamma}^\Gamma t & \implies [s]R_{\beta}^\Gamma [u]R_{\gamma}^\Gamma [t] \\
& \iff [s]R_{\beta;\gamma}^\Gamma [t] \\
& \iff A_s \in t \\
& \implies \mathcal{M}^C, t \models A_s.
\end{align*}
\]

Hence \(\mathcal{M}^C, s \models [\beta][\gamma]A_s\). Using this result we now show that G1 holds:

\[
\begin{align*}
\mathcal{M}^C, s \models [\beta][\gamma]A_s & \implies \mathcal{M}^C, s \models [\beta; \gamma]A_s \\
& \implies sR_{\beta;\gamma}^\Gamma t \implies \mathcal{M}^C, t \models A_s \\
& \implies sR_{\beta;\gamma}^\Gamma t \implies A_s \in t \\
& \implies sR_{\beta;\gamma}^\Gamma t \implies [s]R_{\beta;\gamma}^\Gamma [t].
\end{align*}
\]

Hence G1 holds.

G2: Suppose \([s]R_{\beta;\gamma}^\Gamma t\), and also that \([\beta; \gamma]G \in \Gamma_A\) and \(\mathcal{M}^C, s \models [\beta; \gamma]G\). We need to show that \(\mathcal{M}^C, t \models G\). We first observe the following implications:

\[
\begin{align*}
[s]R_{\beta;\gamma}^\Gamma t & \implies \exists[u][(s]R_{\beta}^\Gamma [u]R_{\gamma}^\Gamma [t]) \quad (3.2) \\
[\beta; \gamma]G \in \Gamma_A & \implies [\beta][\gamma]G \in \Gamma_A \text{ and } [\gamma]G \in \Gamma_A \quad (3.3) \\
\mathcal{M}^C, s \models [\beta; \gamma]G & \begin{array}{c} \text{COMP} \end{array} \mathcal{M}^C, s \models [\beta][\gamma]G. \quad (3.4)
\end{align*}
\]
By the induction hypothesis G2 holds for $R^C_\beta$ and $R^C_\gamma$. Using the above implications we now show that G2 holds for $R^C_{\beta \cup \gamma}$:

$$
[s]R^C_{\beta \cup \gamma}[t] \land [\beta; \gamma]G \in \Gamma_A \land \mathcal{M}^C, s \models [\beta; \gamma]G
$$

$$
\implies \exists [u]([s]R^C_\beta[u]R^C_\gamma[t] \land [\beta][\gamma]G \in \Gamma_A \land [\gamma]G \in \Gamma_A \land \mathcal{M}^C, s \models [\beta][\gamma]G)
$$

$$
\implies \exists [u]([u]R^C_\gamma[t] \land [\gamma]G \in \Gamma_A \land [\beta][\gamma]G \in \Gamma_A \land \mathcal{M}^C, s \models [\beta][\gamma]G))
$$

$$
\implies \exists [u]([u]R^C_\gamma[t] \land [\gamma]G \in \Gamma_A \land \mathcal{M}^C, u \models [\gamma]G)
$$

$$
\implies \mathcal{M}^C, t \models G
$$

Hence G2 holds.

Case 4: $\alpha = \beta \cup \gamma$

G1: Fix $s$ and let $A_s$ be a formula such that for all $t$, $A_s \in t$ if and only if $[s]R^C_{\beta \cup \gamma}[t]$. By the induction hypothesis G1 holds for $R^C_\beta$ and $R^C_\gamma$. We now show that $\mathcal{M}^C, s \models [\beta]A_s \land [\gamma]A_s$. First we need the following implication:

$$
sR^C_\beta t \implies [s]R^C_\beta[t]
$$

$$
\implies [s]R^C_{\beta \cup \gamma}[t]
$$

$$
\implies A_s \in t
$$

$$
\implies \mathcal{M}^C, t \models A_s.
$$

Hence $\mathcal{M}^C, s \models [\beta]A_s$. Similarly $\mathcal{M}^C, s \models [\gamma]A_s$, so $\mathcal{M}^C, s \models ([\beta]A_s \land [\gamma]A_s)$. We now use this to show that G1 holds:

$$
\mathcal{M}^C, s \models ([\beta]A_s \land [\gamma]A_s) \iff \mathcal{M}^C, s \models [\beta \cup \gamma]A_s
$$

$$
sR^C_{\beta \cup \gamma} t \implies \mathcal{M}^C, t \models A_s
$$

$$
\implies sR^C_{\beta \cup \gamma} t \implies A_s \in t
$$

$$
\implies sR^C_{\beta \cup \gamma} t \implies [s]R^C_{\beta \cup \gamma}[t].
$$

Hence G1 holds.
CHAPTER 3. SOUNDNESS AND COMPLETENESS

G2: Suppose $[s]R_{\beta \cup \gamma}^R[t]$, and also that $[\beta \cup \gamma]G \in \Gamma_A$ and $\mathcal{M}^C, s \models [\beta \cup \gamma]G$. We need to show that $\mathcal{M}^C, t \models G$. We require the following implications:

\begin{align}
[s]R_{\beta \cup \gamma}^R[t] & \quad \overset{M2}{\Rightarrow} \quad [s]R_{\beta}^R[t] \text{ or } [s]R_{\gamma}^R[t] \quad (3.5) \\
[\beta \cup \gamma]G \in \Gamma_A & \quad \Rightarrow \quad [\beta]G \in \Gamma_A \text{ and } [\gamma]G \in \Gamma_A \quad (3.6) \\
\mathcal{M}^C, s \models [\beta \cup \gamma]G & \quad \overset{\text{AT}}{\Rightarrow} \quad \mathcal{M}^C, s \models ([\beta]G \land [\gamma]G) \quad (3.7)
\end{align}

By the induction hypothesis G2 holds for $R_{\beta}^R$ and $R_{\gamma}^R$. Using this, and the above implications, we have

\begin{align}
([s]R_{\beta}^R[t] \text{ or } [s]R_{\gamma}^R[t]) \text{ and } [\beta]G \in \Gamma_A \text{ and } [\gamma]G \in \Gamma_A & \quad \Rightarrow \quad (\mathcal{M}^C, s \models [\beta]G \text{ and } \mathcal{M}^C, s \models [\gamma]G) \\
\text{and } \mathcal{M}^C, s \models [\beta \cup \gamma]G & \quad \Rightarrow \quad \mathcal{M}^C, s \models [\beta]G \text{ and } \mathcal{M}^C, s \models [\gamma]G \\
& \quad \Rightarrow \quad (\mathcal{M}^C, s \models [\beta]G \text{ and } \mathcal{M}^C, s \models [\gamma]G) \\
\text{or } ([s]R_{\beta}^R[t] \text{ and } [\gamma]G \in \Gamma_A \text{ and } \mathcal{M}^C, s \models [\gamma]G) & \quad \models \mathcal{M}^C, t \models G.
\end{align}

Hence G2 holds.

Case 5: $\alpha = \beta^*$

G1: Fix $s$ and let $A_s$ be a formula such that for all $t$, $A_s \in t$ if and only if $[s]R_{\beta^*}^R[t]$. We now show that $\mathcal{M}^C, s \models (A_s \rightarrow [\beta^*]A_s)$. We need the following chain of implications:
$\mathcal{M}^C, t \models A_s \quad \overset{3.3.1}{\Rightarrow} \quad A_s \in t$

$\Rightarrow \quad [s]R_{\beta^*}[t]$

$M^3 \quad [s](R_{\beta^*})^*[t]$

$\Rightarrow \quad \exists n([s](R_{\beta^*})^n[t])$

$\Rightarrow \quad \exists n([s](R_{\beta^*})^n[t]) \text{ and } ( tR_{\beta}u \overset{III}{=} [t]R_{\beta}^*[u] )$

$\Rightarrow \quad tR_{\beta}u \Rightarrow ( \exists n([s](R_{\beta^*})^n[t]) \text{ and } [t]R_{\beta}^*[u] )$

$\Rightarrow \quad tR_{\beta}u \Rightarrow \exists n([s](R_{\beta^*})^{n+1}[u])$

$\Rightarrow \quad tR_{\beta}u \Rightarrow ([s](R_{\beta^*})^[u])$

$\Rightarrow \quad tR_{\beta}u \Rightarrow [s]R_{\beta^*}[u]$

$\Rightarrow \quad tR_{\beta}u \Rightarrow A_s \in u$

$\overset{3.3.1}{\Rightarrow} \quad tR_{\beta}u \Rightarrow \mathcal{M}^C, u \models A_s$

$S^6 \quad \mathcal{M}^C, t \models [\beta]A_s.$

Hence $\mathcal{M}^C, t \models A_s \rightarrow [\beta]A_s,$ and this is true for all $t$. In particular if $sR_{\beta^*}t$ then $\mathcal{M}^C, t \models A_s \rightarrow [\beta]A_s,$ so $\mathcal{M}^C, s \models [\beta^*](A_s \rightarrow [\beta]A_s).$ The axiom schema IND yields $\mathcal{M}^C, s \models (A_s \rightarrow [\beta^*]A_s).$ As $[s](R_{\beta^*})^*[s]$ we know from the definition of $R_{\beta^*} = (R_{\beta^*})^*$ that $[s]R_{\beta^*}[s]$. From this we get the following chain of implications:

$[s]R_{\beta^*}[s] \quad \Rightarrow \quad A_s \in s$

$\overset{3.3.1}{\Rightarrow} \quad \mathcal{M}^C, s \models A_s$

$\Rightarrow \quad \mathcal{M}^C, s \models [\beta^*]A_s$

$\overset{S^6}{\Rightarrow} \quad sR_{\beta^*}t \Rightarrow \mathcal{M}^C, t \models A_s$

$\overset{3.3.1}{\Rightarrow} \quad sR_{\beta^*}t \Rightarrow A_s \in t$

$\Rightarrow \quad sR_{\beta^*}t \Rightarrow [s]R_{\beta^*}[t].$

Hence G1 holds.

G2: Suppose that $[s]R_{\beta^*}[t],$ and also that $[\beta^*]G \in \Gamma$ and $\mathcal{M}, s \models [\beta^*]G$. We want to show that $\mathcal{M}, t \models G$. We first use induction to show that for every $n \geq 0$

$$([s]R_{\beta^*}[t] \text{ and } [\beta^*]G \in \Gamma \text{ and } \mathcal{M}, s \models [\beta^*]G) \Rightarrow \mathcal{M}, t \models [\beta^*]G. \quad (3.8)$$
Case \( n = 0 \): Suppose \([s] R_{\beta_0}^\Gamma [t], [\beta^*] G \in \Gamma \) and \( M, s \models [\beta^*] G \). Then \([s] = [t]\), so as 
\([\beta^*] G \in \Gamma \), \( M, s \models [\beta^*] G \) implies \( M, t \models [\beta^*] G \).

Induction step: Assume (3.8) holds for \( n = k \). Then the following chain of implications completes the induction step:

\[
[s] R_{\beta_{k+1}}^\Gamma [t] \text{ and } [\beta^*] G \in \Gamma \text{ and } M, s \models [\beta^*] G
\implies \exists u((s) R_{\beta_k}^\Gamma [u] R_{\beta_k}^\Gamma [t]) \text{ and } [\beta^*] G \in \Gamma \text{ and } M, s \models [\beta^*] G
\implies \exists u((s) R_{\beta_k}^\Gamma [u] \text{ and } [\beta^*] G \in \Gamma \text{ and } M, s \models [\beta^*] G \text{ and } [u] R_{\beta_k}^\Gamma [t])
\overset{(3.8)}{\implies} \exists u(M, u \models [\beta^*] G \text{ and } [\beta^*] G \in \Gamma \text{ and } [u] R_{\beta_k}^\Gamma [t])
\overset{\text{Mix}}{\implies} \exists u([u] R_{\beta_k}^\Gamma [t] \text{ and } [\beta][\beta^*] G \in \Gamma \text{ and } M, u \models [\beta][\beta^*] G)
\overset{\text{IH}}{\implies} M, t \models [\beta^*] G.
\]

Hence (3.8) holds for every \( n \in \mathbb{N} \). We can now show that G2 holds by the following:

\[
[s] R_{\beta_{k+1}}^\Gamma [t] \text{ and } [\beta^*] G \in \Gamma \text{ and } M, s \models [\beta^*] G
\implies \exists u((s) R_{\beta_k}^\Gamma [u] R_{\beta_k}^\Gamma [t]) \text{ and } [\beta^*] G \in \Gamma \text{ and } M, s \models [\beta^*] G
\implies \exists u((s) R_{\beta_k}^\Gamma [u] \text{ and } [\beta^*] G \in \Gamma \text{ and } M, s \models [\beta^*] G)
\implies M, t \models [\beta^*] G
\overset{\text{Mix}}{\implies} M, t \models G.
\]

Hence G2 holds.

\(\square\)

**Lemma 3.3.8** \( \mathcal{M}^\Gamma = (S^\Gamma, \{ R^\Gamma_\alpha : \alpha \in \mathcal{P}^\Gamma \}, V^\Gamma) \) is a standard model for PDL.

**Proof:** We need to show that M1–M4 are satisfied. M1–M3 must hold by definition of \( R^\Gamma_\alpha \).

For M4: for \( F \in \mathcal{P}^\Gamma \), \( R_{F^\gamma} = \{ (\{s\}, \{s\}) : \mathcal{M}^C, s \models F \} \), and by Lemma 3.3.4 \( \mathcal{M}^C, s \models F \) if and only if \( \mathcal{M}^\Gamma, [s] \models F \) for \( F \in \mathcal{P}^\Gamma \). Hence \( R_{F^\gamma} = \{ ([s], [s]) : \mathcal{M}^\Gamma, [s] \models F \} \) and M4 holds for every \( F \in \mathcal{P}^\Gamma \). For \( F \notin \mathcal{P}^\Gamma \) M4 holds by definition of \( R^\Gamma_\alpha \). \(\square\)

**Theorem 3.3.9** PDL is complete with respect to the class of standard models for PDL.
Proof: For any non-theorem \( A \) of PDL let \( \Gamma_A \) denote the smallest FL-closed set containing \( A \). We showed in Lemma 3.3.7 that the model \( M_{\Gamma} \) is a \( \Gamma_A \)-filtration of the canonical model for PDL. As \( A \) is a non-theorem, \( \neg A \) is consistent so there is a maximal consistent set of formulas containing \( \neg A \). Suppose \( \neg A \in s \in S^C \). Then by Lemma 3.3.1 \( M^{C}, s \models \neg A \) so \( M^{C}, s \not\models A \). As \( A \in \Gamma_A \), then by Lemma 3.3.4 \( M^{\Gamma}, [s] \not\models A \). By Lemma 3.3.8 \( M^{\Gamma} \) is a standard model for PDL, hence \( A \) is not valid in every standard model for PDL. Thus the only formulas of PDL which are valid are the theorems of PDL, so PDL is complete with respect to the class of standard models for PDL. □

3.4 Soundness of SPDL

Theorem 3.4.1 SPDL is sound with respect to the class of standard models for SPDL.

Proof: We need to show that each axiom of SPDL is valid in every standard model, and that the transformation rules preserve validity. We only consider the axioms and transformation rules that are not part of PDL. Each of these is treated in turn. Below \( M \) denotes a standard model for SPDL.

DUM:

\[
\begin{align*}
M, s \models [\text{skip}]F & \iff \forall t(s R_{\text{skip}} t \Rightarrow M, t \models F) \\
M^1' & \iff \forall t(s = t \Rightarrow M, t \models F) 
\end{align*}
\]

Hence \( M, s \models [\text{skip}]F \iff F \).

ABORT: By condition M2' we have \( R_{\text{abort}} = \emptyset \). Let \( s \) be a state in a standard SPDL-model. For every state \( t \), \( s R_{\text{abort}} t \) implies \( M, t \models F \), hence \( M, s \models [\text{abort}]F \).
CHAPTER 3. SOUNDNESS AND COMPLETENESS

COND:

\[ M, s \models [\text{if } E \text{ then } \alpha \text{ else } \beta]F \]

\[ \iff \forall t (sR_{\text{if } E \text{ then } \alpha \text{ else } \beta} t \Rightarrow M, t \models F) \]

\[ \iff \forall t(((M, s \models E \text{ and } sR_\alpha t) \text{ or } (M, s \models \neg E \text{ and } sR_\beta t)) \Rightarrow M, t \models F) \]

\[ \iff \forall t(((M, s \models E \text{ and } sR_\alpha t) \Rightarrow M, t \models F) \]

\[ \text{and } ((M, s \models \neg E \text{ and } sR_\beta t) \Rightarrow M, t \models F)) \]

\[ \iff \forall t(((M, s \models E \text{ and } sR_\alpha t) \Rightarrow M, t \models F) \]

\[ \text{and } \forall t((M, s \models \neg E \text{ and } sR_\beta t) \Rightarrow M, t \models F) \]

\[ \iff \forall t(M, s \models E \Rightarrow (sR_\alpha t \Rightarrow M, t \models F)) \text{ and } \forall t(M, s \models \neg E \Rightarrow (sR_\beta t \Rightarrow M, t \models F)) \]

\[ \iff (M, s \models E \Rightarrow \forall t(sR_\alpha t \Rightarrow M, t \models F)) \text{ and } (M, s \models \neg E \Rightarrow \forall t(sR_\beta t \Rightarrow M, t \models F)) \]

\[ \iff (M, s \models E \Rightarrow [\alpha]F) \text{ and } (M, s \models \neg E \Rightarrow [\beta]F) \]

\[ \iff (M, t \models (E \rightarrow [\alpha]F) \wedge (\neg E \rightarrow [\beta]F) \]

Hence \( M, s \models [\text{if } E \text{ then } \alpha \text{ else } \beta]F \iff ((E \rightarrow [\alpha]F) \wedge (\neg E \rightarrow [\beta]F)) \).

\textbf{while1:}

\[ (M, s \models \neg E \iff \forall t(sR_{\text{while } E \text{ do } \alpha} t \Rightarrow s = t) \]

\[ \Rightarrow sR_{\text{while } E \text{ do } \alpha} s \]

\[ \Rightarrow (M, s \models [\text{while } E \text{ do } \alpha]F \Rightarrow M, s \models F) \]

\[ \Rightarrow M, s \models [\text{while } E \text{ do } \alpha]F \rightarrow F. \]

Hence \( M, s \models \neg E \rightarrow [\text{while } E \text{ do } \alpha]F \rightarrow F. \)

\textbf{while2:} Suppose \( M, s \models [\text{while } E \text{ do } \alpha]F \) and \( M, s \models E \). We need to show that \( M, s \models [\alpha][\text{while } E \text{ do } \alpha]F \). Let \( t \) be such that \( sR_\alpha t \). Then for every \( u \) such that \( tR_{\text{while } E \text{ do } \alpha} u \) we have \( sR_{\text{while } E \text{ do } \alpha} u \) so \( M, u \models F \), and hence \( M, t \models [\text{while } E \text{ do } \alpha]F \). Hence \( M, t \models [\text{while } E \text{ do } \alpha]F \) for every \( t \) such that \( sR_\alpha t \), and so as required we get \( M, s \models [\alpha][\text{while } E \text{ do } \alpha]F \).
CHAPTER 3. SOUNDNESS AND COMPLETENESS

Hoare's Iteration Rule: Suppose $\mathcal{M} \models (E \land F) \rightarrow [\alpha]F$. Let $s$ be any state such that $\mathcal{M}, s \models F$. We show that $\mathcal{M}, s \models [\text{while } E \text{ do } \alpha]F$. Let $t$ be such that $s R_{\text{while } E \text{ do } \alpha} t$. Then there exist $n \in \mathbb{N}$ and $s_0, s_1, \ldots, s_n$ such that $s = s_0$, $t = s_n$, $\mathcal{M}, t \models \neg E$, and for all $j < n$ we have $s_j R_{\alpha} s_{j+1}$ and $\mathcal{M}, s_j \models E$. By induction we show that $\mathcal{M}, s_j \models F$ for all $j$, $0 \leq j \leq n$. We are given $\mathcal{M}, s_0 \models F$ so the base case holds. Now suppose $\mathcal{M}, s_k \models F$ for some $k < n$. We also know $\mathcal{M}, s_k \models E$, hence $\mathcal{M}, s_k \models (E \land F)$ so $\mathcal{M}, s_j \models [\alpha]F$. As $s_k R_{\alpha} s_{k+1}$ it follows that $\mathcal{M}, s_{j+1} \models F$ as required. Hence $s R_{\text{while } E \text{ do } \alpha} t$ implies $\mathcal{M}, t \models F$, so $\mathcal{M}, s \models [\text{while } E \text{ do } \alpha]F$. Thus $\mathcal{M}, s \models F \rightarrow [\text{while } E \text{ do } \alpha]F$ for every $s \in S$, so $\mathcal{M} \models F \rightarrow [\text{while } E \text{ do } \alpha]F$ and Hoare's Iteration Rule holds for $\mathcal{M}$.

3.5 Completeness of SPDL

In this section we will show that SPDL is complete with respect to the class of standard SPDL models. As for PDL we show that every non-theorem of SPDL is not valid in some standard SPDL model. We first construct the canonical model. The canonical model for SPDL is the model $\mathcal{M}^C = (S^C, \{R^C_{\alpha} : \alpha \in \mathcal{P}_{\text{SPDL}}\}, V^C)$ where

- $S^C$ is the set of all maximal SPDL-consistent sets

- For each $\alpha \in \mathcal{P}_{\text{SPDL}}$, $R^C_{\alpha}$ is the binary relation defined by $s R^C_{\alpha} t \iff \{G \in F : [\alpha] G \in s\} \subseteq t$

- $V^C : \emptyset \rightarrow \mathcal{P}(S^C)$ is the function given by $V^C(p) = \{s \in S^C : p \in s\}$.

Lemma 3.5.1 Let $F$ be a formula of SPDL. Then for every $s \in S^C$, $\mathcal{M}^C, s \models F \iff F \in s$.

Proof: By induction on formulas. □

It follows that every theorem of SPDL is valid in $\mathcal{M}^C$, and no non-theorem of SPDL is valid in $\mathcal{M}^C$. However $\mathcal{M}^C$ is not necessarily a standard model of SPDL. We now construct for each non-theorem $A$ of SPDL a standard model in which $A$ is not valid.
Definition 3.5.2 A set $\Gamma$ of SPDL-formulas is called $FLS$-closed if it is closed under subformulas and the following conditions hold for all $F, G \in \mathcal{F}_{SPDL}$ and $\alpha, \beta \in \mathcal{P}_{SPDL}$:

(i) $[\alpha; \beta]F \in \Gamma \Rightarrow [\alpha][\beta]F \in \Gamma$

(ii) $[\text{while } E \text{ do } \alpha]F \in \Gamma \Rightarrow [\alpha]([[\text{while } E \text{ do } \alpha]]F) \in \Gamma$

(iii) $[\text{if } E \text{ then } \alpha \text{ else } \beta]F \in \Gamma \Rightarrow [\alpha]F \in \Gamma$ and $[\beta]F \in \Gamma$.

Let $\Gamma$ be a $FLS$-closed set of SPDL formulas. As for PDL we define the notions of a $(\Gamma, \alpha)$-filtration of a binary relation, and a $\Gamma$-filtration of a model.

Lemma 3.5.3 Suppose that $\mathcal{M}^\Gamma = (S^\Gamma, \{R^\alpha_\Gamma : \alpha \in \mathcal{P}^\Gamma\}, V^\Gamma)$ is a $\Gamma$-filtration of the model $\mathcal{M} = (S, \{R_\alpha : \alpha \in \mathcal{P}\}, V)$ such that $\Gamma$ is finite, and let $T$ be a subset of $S^\Gamma$. Then there is a formula $A_T$ such that $A_T \in s$ if and only if $[s] \in T$.

Proof: As for PDL. $\Box$

Let $A$ be a non-theorem of SPDL. Let $\Gamma_A$ be the smallest $FLS$-closed set containing $A$. As before we can show that $\Gamma_A$ is finite. Again note that $|S^\Gamma| \leq 2^{\lvert \Gamma_A \rvert}$, hence it follows that $S^\Gamma$ is finite, and we can apply Lemma 3.5.3.

Lemma 3.5.4 $\Gamma_A$ is finite.

Proof: The proof is essentially as for PDL. We need the following observations for conditions (ii) and (iii):

(ii) Every subformula of $[\alpha][\text{while } E \text{ do } \alpha]F$, except $[\alpha][\text{while } E \text{ do } \alpha]F$ itself, is a subformula of $[\text{while } E \text{ do } \alpha]F$. 
(iii) Every subformula of \([a]F\) or \([\beta]F\), except for \([\alpha]F\) and \([\beta]F\), is a subformula of \([\text{if } E \text{ then } \alpha \text{ else } \beta]F\).

\[ \square \]

For the rest of this section let \(M^\Gamma\) denote the model \(\langle S^\Gamma, \{R^\alpha : \alpha \in \mathcal{P}^\Gamma\}, V^\Gamma \rangle\), where \(\Gamma = \Gamma_A\) and

- \(R^\Gamma_{\pi}\) is a \(\Gamma_A\)-filtration of \(R^C_{\pi}\) for \(\pi \in \Pi^\Gamma_A\)
- \(R^\Gamma_{\pi}\) is arbitrary for \(\pi \in \Pi \setminus \Pi^\Gamma_A\)
- \(R^\Gamma_{\text{skip}} = id_{S^\Gamma}\)
- \(R^\Gamma_{\text{abort}} = \emptyset\)
- \(R^\Gamma_\alpha\) is defined by the standard model conditions for SPDL otherwise.

We will prove simultaneously by induction that for every formula \(F \in \Gamma_A, M^C, s \models F\) if and only if \(M^\Gamma, [s] \models F\), and that \(R^\Gamma_\alpha\) is a \((\Gamma, \alpha)\)-filtration of \(R^C_\alpha\) for every \(\alpha \in \mathcal{P}^\Gamma\). To do this we need to define the following sets:

- \(\mathcal{P}_0\) is the set of programs in \(\mathcal{P}^\Gamma\) that contain no \textbf{while} or \textbf{if} statements.
- \(\mathcal{F}_n\) is the set of formulas of \(\Gamma_A\) that contain programs only from \(\mathcal{P}_n\).
- \(\mathcal{P}_{n+1}\) is the set of programs in \(\mathcal{P}^\Gamma\) such that any statement of the form "\textbf{while } E \textbf{ do } \beta" or "\textbf{if } E \textbf{ then } \beta \textbf{ else } \gamma" has \(E \in \mathcal{F}_n\) and \(\beta, \gamma \in \mathcal{P}_n\).

We observe that \(\mathcal{P}^\Gamma = \bigcup_{n \in \mathbb{N}} \mathcal{P}_n\), and \(\Gamma_A = \bigcup_{n \in \mathbb{N}} \mathcal{F}_n\).

**Theorem 3.5.5** For every formula \(F \in \Gamma_A, M^C, s \models F\) if and only if \(M^\Gamma, [s] \models F\), and for every program \(\alpha \in \mathcal{P}^\Gamma\), \(R^\Gamma_\alpha\) is a \((\Gamma, \alpha)\)-filtration of \(R^C_\alpha\).

**Proof:** It suffices to prove that for every \(n \in \mathbb{N}\) the following hold:

(i) for every \(\alpha \in \mathcal{P}_n\), \(R^\Gamma_\alpha\) is a \((\Gamma, \alpha)\)-filtration of \(R^C_\alpha\)
(ii) for every \(F \in \mathcal{F}_n, M^C, s \models F\) if and only if \(M^\Gamma, [s] \models F\).

We use strong induction on \(n\).

Base step: \(n = 0\). For (i) we use induction on the construction of programs in \(\mathcal{P}_0\).
CHAPTER 3. SOUNDNESS AND COMPLETENESS

Case 1: $\alpha \in \Pi^\Gamma$. Then $R^\Gamma_\alpha$ is a $\Gamma_A$-filtration of $R^C_\alpha$ by definition of $R^\Gamma_\alpha$.

Case 2: $\alpha = \text{skip}$.

G1: We need the following implication:

\[
\begin{align*}
{s R^C_{\text{skip}} t} & \iff \{ F \in \mathcal{F}_{\text{SPDL}} : [\text{skip}]F \in s \} \subseteq t \\
& \overset{\text{Dum}}{\iff} \{ F \in \mathcal{F}_{\text{SPDL}} : F \in s \} \subseteq t \\
& \implies s \subseteq t.
\end{align*}
\]

Hence $s = t$ as both $s$ and $t$ are maximal. By definition $[s]R^\Gamma_{\text{skip}}[s]$ hence $[s]R^\Gamma_{\text{skip}}[t]$ as required, and G1 holds.

G2: This is proved as follows:

\[
\begin{align*}
[s]R^\Gamma_{\text{skip}}[t] \text{ and } \mathcal{M}, s \models [\text{skip}]F \text{ and } [\text{skip}]F \in \Gamma_A \\
\implies s \sim t \text{ and } \mathcal{M}, s \models [\text{skip}]F \text{ and } [\text{skip}]F \in \Gamma_A \\
\implies \mathcal{M}, t \models [\text{skip}]F \\
\overset{\text{Dum}}{\iff} \mathcal{M}, t \models F.
\end{align*}
\]

Case 3: $\alpha = \text{abort}$.

G1: Suppose $sR^C_{\text{abort}}t$. Then $\{ F \in \mathcal{F}_{\text{SPDL}} : [\text{abort}]F \in s \} \subseteq t$. This implies false $\in t$, hence there is no $t$ such that $sR^C_{\text{abort}}t$, and trivially $sR^C_{\text{abort}}t$ implies $[s]R^\Gamma_{\text{abort}}[t]$.

G2: There does not exist $t$ such that $[s]R^\Gamma_{\text{abort}}[t]$, hence trivially $([s]R^\Gamma_{\text{abort}}[t] \text{ and } [\text{abort}]F \in \Gamma_A \text{ and } \mathcal{M}^C, s \models [\text{abort}]F) \implies \mathcal{M}^C, t \models F$.

Case 4: $\alpha = \beta; \gamma$. This is proved as in Lemma 3.3.7.

For (ii) we use induction on the construction of formulas in $\mathcal{F}_0$, which are formulas that contain no programs containing while or if statements.

Case 1: $F = p \in \Phi^\Gamma$. Then

\[
\begin{align*}
\mathcal{M}^C, s \models p & \iff s \in V(p) \\
& \iff [s] \in V^\Gamma(p) \\
& \overset{\text{S1}}{\iff} \mathcal{M}^\Gamma, [s] \models p.
\end{align*}
\]
Case 2: \( F = \text{true} \). From S2 we have both \( M^C, s \models \text{true} \) and \( M^\Gamma, [s] \models \text{true} \) hence \( M^C, s \models \text{true} \) if and only if \( M^\Gamma, [s] \models \text{true} \).

Case 3: \( F = \text{false} \). From S3 we have both \( M^C, s \not\models \text{false} \) and \( M^\Gamma, [s] \not\models \text{false} \) hence \( M^C, s \models \text{false} \) if and only if \( M^\Gamma, [s] \models \text{false} \).

Case 4: \( F = \neg G \). Then

\[
M^C, s \models F \iff M^C, s \not\models G \\
\iff M^\Gamma, [s] \not\models G \\
\iff M^\Gamma, [s] \models F.
\]

Case 5: \( F = G \lor H \). Then

\[
M^C, s \models F \iff M^C, s \models G \, \text{or} \, M^C, s \models H \\
\iff M^\Gamma, [s] \models G \, \text{or} \, M^\Gamma, [s] \models H \\
\iff M^\Gamma, [s] \models F.
\]

Case 6: \( F = [\alpha]G \), where \( \alpha \in \mathcal{P}_0 \) and by induction \( G \in \mathcal{F}_0 \). From (i) \( R^\Gamma_\alpha \) is a \((\Gamma_A, \alpha)\)-filtration of \( R_\alpha \). We prove each direction of the equivalence separately.

For the first suppose that \( M^C, s \models [\alpha]G \). We know \( [\alpha]G \in \Gamma_A \), so it follows from G2 that \([s]R^\Gamma_\alpha [t] \) implies \( M^C, t \models G \), which by the induction hypothesis for \( G \) implies \( M^\Gamma, [t] \models G \). Hence \( M^\Gamma, [s] \models [\alpha]G \) as required.

For the other direction suppose \( M^\Gamma, [s] \models [\alpha]G \). We need the following chain of implications:

\[
sR^C_\alpha t \quad \Rightarrow \quad [s]R^\Gamma_\alpha [t] \\
\Rightarrow \quad M^\Gamma, [t] \models G \\
\Rightarrow \quad M^C, t \models G.
\]

Hence \( M^C, s \models [\alpha]G \).

Induction step: Assume (i) and (ii) hold for \( \mathcal{P}_i \) and \( \mathcal{F}_i \) respectively for every \( i \leq n \). We first show that (i) holds for \( \mathcal{P}_{n+1} \). There are two cases we need to consider (the remaining cases are proved as for the base case):
Case 1: $\alpha = \text{while } E \text{ do } \beta$, where $E \in \mathcal{F}_n$ and $\beta \in \bigcup_{i \leq n} P_i$.

G1: Let $A_s$ be a formula such that $A_s \in t$ if and only if there is an $m \in N$ such that $\langle s \rangle (E \mid R_\beta)^m[t]$, where $(E \mid R_\beta) = \{([s], [t]) : [s] R_\beta[t] \text{ and } M^\Gamma, [s] \models E\}$ (such an $A_s$ exists by Lemma 3.5.3). We now show that $M^C, [E \land A_s] \models [\beta] A_s$. Let $u$ be any state such that $M^C, u \models E \land A_s$, and suppose $u R_\beta t$.

\[
M^C, u \models (E \land A_s) \text{ and } u R_\beta t
\]

\[
\models_{\text{IH}} M^C, u \models A_s \text{ and } M^C, u \models E \text{ and } [u] R_\beta[t]
\]

\[
3.5.1 \text{IH} \quad A_s \in u \text{ and } M^\Gamma, [u] \models E \text{ and } [u] R_\beta[t]
\]

\[
\iff \exists m([s](E \mid R_\beta)^m[u]) \text{ and } M^\Gamma, [u] \models E \text{ and } [u] R_\beta[t]
\]

\[
\iff (\exists m)([s](E \mid R_\beta)^m[u]) \text{ and } [u](E \mid R_\beta)[t]
\]

\[
\iff \exists m([s](E \mid R_\beta)^{m+1}[t])
\]

\[
\iff A_s \in t
\]

\[
3.5.1 \quad M^C, t \models A_s.
\]

Hence

\[
M^C, u \models (E \land A_s) \quad \implies \quad u R_\beta t \implies M^C, t \models A_s
\]

\[
\implies \quad M^C, u \models [\beta] A_s.
\]

Hence $M, u \models (E \land A_s) \rightarrow [\beta] A_s$ for every state $u$, so $M \models (E \land A_s) \rightarrow [\beta] A_s$. By Hoare's iteration rule we get $M \models A_s \rightarrow [\text{while } E \text{ do } \beta](A_s \land \neg E)$. As $[s](E \mid R_\beta)^0[s]$ we have $A_s \in s$, so by Lemma 3.5.1 $M^C, s \models A_s$ and so $M^C, s \models [\text{while } E \text{ do } \beta](A_s \land \neg E)$. Hence

\[
s R_{\text{while } E \text{ do } \beta} t \quad \implies \quad M^C, t \models A_s \land \neg E
\]

\[
\iff \quad M^C, t \models A_s \text{ and } M^C, t \models \neg E
\]

\[
3.5.1 \quad A_s \in t \text{ and } M^C, t \models \neg E
\]

\[
\iff \exists m([s](E \mid R_\beta)^m[t]) \text{ and } M^C, t \models \neg E
\]

\[
\models_{\text{IH}} \exists m([s](E \mid R_\beta)^m[t]) \text{ and } M^\Gamma, [t] \models \neg E
\]

\[
M^C \quad [s] R_{\text{while } E \text{ do } \beta} t.
\]

Hence G1 holds.
G2: Suppose \([s]R_{\text{while } E \text{ do } \beta}[t]\), \([\text{while } E \text{ do } \beta]G \in \Gamma_A\) and \(\mathcal{M}^C, s \models [\text{while } E \text{ do } \beta]G\). Then, as \([s]R_{\text{while } E \text{ do } \beta}[t]\), there exist \(m \in \mathbb{N}\) and \(s_0, s_1, \ldots, s_m \in S\) such that \([s_0] = [s]s_m = [t]\), \(\mathcal{M}^\Gamma, [s_m] \models \neg E\) and for every \(j \leq m\), \([s_j]R_\beta[s_{j+1}]\) and \(\mathcal{M}^\Gamma, [s_j] \models E\). We now use induction to show that \(\mathcal{M}^C, s_i \models [\text{while } E \text{ do } \beta]G\) for every \(i \leq m\).

Base step: \(i = 0\). This is true by hypothesis.

Induction step: Assume \(\mathcal{M}^C, s_k \models [\text{while } E \text{ do } \beta]G\), for some \(k < m\). As \(\mathcal{M}^\Gamma, [s_k] \models E\), and \(E \in \mathcal{F}_n\) we have \(\mathcal{M}^C, s_k \models E\). From the axiom schema \([\text{WHILE}2]\) we have \(\mathcal{M}^C, s_k \models [\beta][\text{while } E \text{ do } \beta]G\). As \(\Gamma_A\) is \(\mathcal{F}_S\)-closed, \([\beta][\text{while } E \text{ do } \beta]G \in \Gamma_A\). We also know \([s_k]R_\beta^\Gamma[s_{k+1}]\). As \(\beta \in \mathcal{P}_n\) we know \(R_\beta^\Gamma\) is a \((\Gamma, \beta)\)-filtration of \(R_\beta^G\) and G2 holds. This yields \(\mathcal{M}^C, s_{k+1} \models [\text{while } E \text{ do } \beta]G\).

Hence \(\mathcal{M}^C, s_m \models [\text{while } E \text{ do } \beta]G\). From above \(\mathcal{M}^\Gamma, [s_m] \models \neg E\). As \(\neg E \in \mathcal{F}_n\), we have \(\mathcal{M}^C, s_m \models \neg E\). The axiom schema \([\text{WHILE}1]\) yields \(\mathcal{M}^C, s \models G\), which is what was required to show G2 holds.

Case 2: \(\alpha = \text{if } E \text{ then } \beta \text{ else } \gamma\)

G1: Let \(A_s\) be a formula such that \(A_s \in t\) iff \([s]R^G_{\text{if } E \text{ then } \beta \text{ else } \gamma}[t]\). We now show that \(\mathcal{M}^C, s \models (E \rightarrow [\beta]A_s) \land (\neg E \rightarrow [\gamma]A_s)\).

\[
\mathcal{M}^C, s \models E \quad \Rightarrow \quad \mathcal{M}^\Gamma, s \models E
\]

\[
\quad \Rightarrow \quad sR^C_\beta t \Rightarrow ([s]R^\Gamma_\beta[t]\text{ and } \mathcal{M}^\Gamma, s \models E)
\]

\[
\quad \Rightarrow \quad sR^C_\alpha t \Rightarrow [s]R^\Gamma_{\text{if } E \text{ then } \beta \text{ else } \gamma}[t]
\]

\[
\Rightarrow \quad sR^C_\beta t \Rightarrow A_s \in t
\]

\[
\Rightarrow \quad sR^C_\beta t \Rightarrow \mathcal{M}^C, t \models A_s
\]

\[
\Rightarrow \quad \mathcal{M}^C, s \models [\beta]A_s.
\]

Hence \(\mathcal{M}^C \models E \rightarrow [\beta]A_s\). Similarly we can show that \(\mathcal{M}^C \models \neg E \rightarrow [\gamma]A_s\). Thus \(\mathcal{M}^C \models (E \rightarrow [\beta]A_s) \land (\neg E \rightarrow [\gamma]A_s)\). The axiom schema \([\text{COND}]\) yields \(\mathcal{M}^C, s \models [\text{if } E \text{ then } \beta \text{ else } \gamma]A_s\). We can now use this to show G1 holds:

\[
\mathcal{M}^C, s \models [\text{if } E \text{ then } \beta \text{ else } \gamma]A_s \quad \Leftrightarrow \quad sR^C_{\text{if } E \text{ then } \beta \text{ else } \gamma}[t] \Rightarrow \mathcal{M}^C, t \models A_s
\]

\[
\quad \Rightarrow \quad sR^C_{\text{if } E \text{ then } \beta \text{ else } \gamma}[t] \Rightarrow A_s \in t
\]

\[
\quad \Leftrightarrow \quad sR^C_{\text{if } E \text{ then } \beta \text{ else } \gamma}[t] \Rightarrow [s]R^\Gamma_{\text{if } E \text{ then } \beta \text{ else } \gamma}[t].
\]
Hence G1 holds.

G2: Suppose that \([s]R^\Gamma_{if E then \beta else \gamma}[t]\), and also that \([if E then \beta else \gamma]G \in \Gamma_A\) and \(M, s \models [if E then \beta else \gamma]G\). We want to show that \(M, t \models G\). We need the following implications:

\[ \begin{align*}
[s]R^\Gamma_{if E then \beta else \gamma}[t] &\quad \stackrel{M5'}{\iff} \quad (\mathcal{M}^\Gamma, [s] \models E \text{ and } [s]R^\Gamma_{\beta}[t]) \\
&\quad \text{or } (\mathcal{M}^\Gamma, [s] \models \neg E \text{ and } [s]R^\Gamma_{\gamma}[t]) \\
\quad \text{(3.9)} \\
\end{align*} \]

\[ \begin{align*}
[if E then \beta else \gamma]G \in \Gamma_A &\quad \implies \quad [\beta]G \in \Gamma_A \text{ and } [\gamma]G \in \Gamma_A \\
\quad \text{(3.10)} \\
\mathcal{M}, s \models [if E then \beta else \gamma]G &\quad \text{COND} \quad \iff \quad \mathcal{M}^C, s \models (E \rightarrow [\beta]G) \land (\neg E \rightarrow [\gamma]G) \\
&\quad \iff \quad \mathcal{M}^C, s \models E \rightarrow [\beta]G \\
&\quad \text{and } \mathcal{M}^C, s \models \neg E \rightarrow [\gamma]G. \\
\quad \text{(3.11)} \\
\end{align*} \]

We now use these implications to show that G2 holds:

\[ \begin{align*}
(s)R^\Gamma_{if E then \beta else \gamma}[t] \text{ and } [if E then \beta else \gamma]G \in \Gamma_A \\
\text{and } \mathcal{M}, s \models [if E then \beta else \gamma]G \\
\quad \text{(3.9)(3.10)(3.11)} \\
\end{align*} \]

\[ \begin{align*}
(\mathcal{M}^\Gamma, [s] \models E \text{ and } [s]R^\Gamma_{\beta}[t]) \text{ or } (\mathcal{M}^\Gamma, [s] \models \neg E \text{ and } [s]R^\Gamma_{\gamma}[t]) \\
\text{and } [\beta]G \in \Gamma_A \text{ and } [\gamma]G \in \Gamma_A \text{ and } \mathcal{M}^C, s \models E \rightarrow [\beta]G \\
\text{and } \mathcal{M}^C, s \models \neg E \rightarrow [\gamma]G \\
\end{align*} \]

\[ \begin{align*}
(\mathcal{M}^C, s \models E \text{ and } [s]R^\Gamma_{\beta}[t]) \text{ or } (\mathcal{M}^C, s \models \neg E \text{ and } [s]R^\Gamma_{\gamma}[t]) \\
\text{and } [\gamma]G \in \Gamma_A \text{ and } \mathcal{M}^C, s \models E \rightarrow [\beta]G \text{ and } \mathcal{M}^C, s \models \neg E \rightarrow [\gamma]G \\
\end{align*} \]

\[ \begin{align*}
(\mathcal{M}^C, s \models \neg E \text{ and } [s]R^\Gamma_{\gamma}[t]) \text{ and } [\gamma]G \in \Gamma_A \text{ and } \mathcal{M}^C, s \models E \rightarrow [\beta]G \\
\text{or } (\mathcal{M}^C, s \models E \rightarrow [\beta]G) \\
\end{align*} \]

\[ \begin{align*}
([s]R^\Gamma_{\beta}[t] \text{ and } [\beta]G \in \Gamma_A \text{ and } \mathcal{M}^C, s \models [\beta]G) \\
\text{or } ([s]R^\Gamma_{\gamma}[t] \text{ and } [\gamma]G \in \Gamma_A \text{ and } \mathcal{M}^C, s \models [\gamma]G) \\
\end{align*} \]

\[ \begin{align*}
\mathcal{M}^C, t \models G. \\
\end{align*} \]

Hence G2 holds.

Hence (i) holds for \(P_{n+1}\). The proof that (ii) holds for \(F_{n+1}\) is the same as for the base case. 
\Box
Lemma 3.5.6 $M^\Gamma = (S^\Gamma, \{R^\Gamma_\alpha : \alpha \in P^\Gamma\}, V^\Gamma)$ is a standard model for SPDL.

**Proof:** We need to show that M1 and M2'-M5' are satisfied. These all hold by definition of $R^\Gamma_\alpha$. $\square$

Theorem 3.5.7 SPDL is complete with respect to the class of standard models for SPDL.

**Proof:** For any non-theorem $A$ of SPDL we let $\Gamma_A$ denote the smallest FLs-closed set containing $A$. We showed in Lemma 3.5.5 that the model $M^\Gamma$ is a $\Gamma_A$ filtration of the canonical model for SPDL. As $A$ is a non-theorem, $\neg A$ is consistent so there is a maximal consistent set of formulas containing $\neg A$. Suppose $\neg A \in s \in S^C$. Then by Lemma 3.5.1 $M^C, s \models \neg A$, so $M^C, s \not\models A$. As $A \in \Gamma_A$, then by Lemma 3.5.5 $M^\Gamma, [s] \not\models A$. By Lemma 3.5.6 $M^\Gamma$ is a standard model for SPDL, hence $A$ is not a valid. Thus the only formulas of SPDL which are valid are the theorems of SPDL, so SPDL is complete with respect to the class of standard models for SPDL. $\square$

3.6 Soundness and Completeness of SPDL+$U$ and SPDL+$*$

It follows from the proofs of Theorems 3.2.1 and 3.4.1 that SPDL+$U$ and SPDL+$*$ are sound with respect to the appropriate classes of standard models. Completeness follows from the proofs of Theorems 3.3.9 and 3.5.7.
Chapter 4

Expressive Power

4.1 Definitions

For this section we use \( \mathcal{L} \) to denote one of the logics PDL, SPDL, SPDL+\( \cup \), and SPDL+\( \ast \), and when we refer to a "logic" we mean one of these logics.

Given a nonempty set \( S \), a collection \( \{ R_\alpha : \alpha \in \Pi \} \) of binary relations, and a function \( V : \Phi \to \mathcal{P}(S) \) we can use the standard model conditions for \( \mathcal{L} \) to define a unique set of relations \( \{ R_\alpha : \alpha \in \mathcal{P}_\mathcal{L} \setminus \Pi \} \) such that \( \mathcal{M} = \langle S, \{ R_\alpha : \alpha \in \mathcal{P}_\mathcal{L} \}, V \rangle \) is a standard model for \( \mathcal{L} \). Thus we could have defined a model (for any of the logics) to be a structure \( \mathcal{M} = \langle S, \{ R_\alpha : \alpha \in \Pi \}, V \rangle \), and defined relations \( \{ R_\alpha : \alpha \in \mathcal{P}_\mathcal{L} \setminus \Pi \} \) using the standard model conditions for \( \mathcal{L} \). In this section it will be convenient to suppose that to have been done. Hence \( \mathcal{M} = \langle S, \{ R_\alpha : \alpha \in \Pi \}, V \rangle \) is a standard model for each of the four logics we are considering. We write \( \mathcal{M}, s \models \mathcal{L} F \) to denote that a formula \( F \) of the logic \( \mathcal{L} \) is true at state \( s \) in \( \mathcal{M} \).

**Definition 4.1.1** Let \( F_1 \) be a formula of a logic \( \mathcal{L}_1 \) and \( F_2 \) be a formula of a logic \( \mathcal{L}_2 \). We say \( F_1 \) and \( F_2 \) are equivalent, written \( F_1 \equiv F_2 \), if for every model \( \mathcal{M} = \langle S, \{ R_\alpha : \alpha \in \Pi \}, V \rangle \) and every \( s \in S \) we have \( \mathcal{M}, s \models \mathcal{L}_1 F_1 \) if and only if \( \mathcal{M}, s \models \mathcal{L}_2 F_2 \).

**Definition 4.1.2** Let \( \alpha_1 \) be a program of a logic \( \mathcal{L}_1 \) and \( \alpha_2 \) be a program of a logic \( \mathcal{L}_2 \). We say \( \alpha_1 \) and \( \alpha_2 \) are equivalent, written \( \alpha_1 \equiv \alpha_2 \), if for every model \( \mathcal{M} = \langle S, \{ R_\alpha : \alpha \in \Pi \}, V \rangle \) we have \( R_{\alpha_1} = R_{\alpha_2} \).
CHAPTER 4. EXPRESSIVE POWER

Definition 4.1.3 Let $L_1$, $L_2$ be logics. We say that a formula $F$ (or program $\alpha$) of $L_1$ can be expressed in $L_2$ if there is a formula $F'$ (or program $\alpha'$) of $L_2$ which is equivalent to $F$ (or $\alpha$).

Definition 4.1.4 We say $L_1$ and $L_2$ have equivalent expressive power if every formula of $L_1$ can be expressed in $L_2$, and every formula of $L_2$ can be expressed in $L_1$. We say that $L_1$ has greater expressive power than $L_2$ if every formula of $L_2$ can be expressed in $L_1$ but there is a formula of $L_1$ which cannot be expressed in $L_2$. In this case $L_2$ has less expressive power than $L_1$.

4.2 Expressive Power of PDL and SPDL

We now compare the expressive powers of PDL and SPDL. We will show that PDL has greater expressive power than SPDL. We first show that every formula of SPDL can be expressed in PDL, then give a formula of PDL which cannot be expressed in SPDL. We will in fact prove that every formula and also every program of SPDL can be expressed in PDL. Every program of SPDL is deterministic, hence any non-deterministic program of PDL, such as $\sigma \cup \tau$ or $\sigma^*$, cannot be expressed in SPDL.

Lemma 4.2.1 Every formula and program of SPDL can be expressed in PDL.

Proof: By induction on the construction of formulas and programs. We only show the cases for programs, as the cases for formulas are trivial. Let $\alpha$ be a program of SPDL. We consider the possibilities for $\alpha$:

Case 1: $\alpha \in \Pi$. Then $\alpha$ is a program of PDL.

Case 2: $\alpha = \text{skip}$. Let $\alpha' = \text{true} \in \mathcal{P}_{\text{PDL}}$. Then

$$R_{\alpha'} = \{(s, s) : M, s \models_{\text{PDL}} \text{true}\}$$
$$= \{(s, s) : s \in S\}$$
$$= \text{id}_S$$
$$= R_{\text{skip}}$$
$$= R_{\alpha}$$
CHAPTER 4. EXPRESSIVE POWER

Case 3: \( \alpha = \text{abort} \). Let \( \alpha' = \text{false} \) \( \in \mathcal{P}_{\text{PDL}} \). Then

\[
R_{\alpha'} = \{(s, s) : M, s \models_{\text{PDL}} \text{false}\}
= \emptyset
= R_{\text{abort}}
= R_{\alpha}
\]

Case 4: \( \alpha = \beta; \gamma \). By induction there exist programs \( \beta' \) and \( \gamma' \) of PDL which are equivalent to \( \beta \) and \( \gamma \) respectively. Then \( \alpha' = \beta'; \gamma' \) is a program of PDL which is equivalent to \( \alpha \).

Case 5: \( \alpha = \text{while } E \text{ do } \beta \). By induction there exist a formula \( E' \) and a program \( \beta' \) of PDL which are equivalent to \( E \) and \( \beta \) respectively. Let \( \alpha' = (E' ?; \beta')^*; \neg E' ? \). We have

\[
R_{E'} = \{(s, s) : M, s \models_{\text{PDL}} E'\}
= \{(s, s) : M, s \models_{\text{PDL}} E\}
\]

\[
R_{E', \beta'} = \{(s, s') : (s, s) \in R_{E'} \text{ and } (s, s') \in R_{\beta'}\}
= \{(s, s') : M, s \models_{\text{PDL}} E \text{ and } (s, s') \in R_{\beta}\}
\]

\[
R_{(E', \beta')}^* = \{(s, s') : (\exists n)(\exists s_0, . . . , s_n \in S)(s = s_0, s' = s_n, \text{ and } \\
(\forall j < n)((s_j, s_{j+1}) \in R_{(E', \beta')}))\}
= \{(s, s') : (\exists n)(\exists s_0, . . . , s_n \in S)(s = s_0, s' = s_n, \text{ and } \\
(\forall j < n)(M, s_j \models_{\text{PDL}} E \text{ and } (s_j, s_{j+1}) \in R_{\beta}))\}
\]

\[
R_{\alpha'} = R_{(E', \beta')}^*; \neg E'
= \{(s, s') : (\exists n)(\exists s_0, . . . , s_n \in S)(s = s_0, s' = s_n, \text{ and } \\
(\forall j < n)(M, s_j \models_{\text{PDL}} E \text{ and } (s_j, s_{j+1}) \in R_{\beta} \text{ and } (s', s') \in R_{\neg E'}))\}
= \{(s, s') : (\exists n)(\exists s_0, . . . , s_n \in S)(s = s_0, s' = s_n, \text{ and } \\
(\forall j < n)(M, s_j \models_{\text{PDL}} E \text{ and } (s_j, s_{j+1}) \in R_{\beta} \text{ and } M, s' \models_{\text{PDL}} \neg E))\}
= R_{\text{while } E \text{ do } \beta}
= R_{\alpha}
\]
Case 6: \( \alpha = \text{if } E \text{ then } \beta \text{ else } \gamma \). By induction there exist a formula \( E' \) and programs \( \beta' \) and \( \gamma' \) of PDL which are equivalent to \( E, \beta \) and \( \gamma \) respectively. Let \( \alpha' = (E'?; \beta') \cup (\neg E'?; \gamma') \). Then

\[
R_{\alpha'} = R_{E'?; \beta'} \cup R_{\neg E'?; \gamma'}
\]

\[
= \{(s, s') : M, s \models_{\text{PDL}} E' \text{ and } (s, s') \in R_{\beta'}\}
\cup \{(s, s') : M, s \models_{\text{PDL}} \neg E' \text{ and } (s, s') \in R_{\gamma'}\}
\]

\[
= \{(s, s') : M, s \models_{\text{SPDL}} E \text{ and } (s, s') \in R_{\beta}\}
\cup \{(s, s') : M, s \models_{\text{SPDL}} \neg E \text{ and } (s, s') \in R_{\gamma}\}
\]

\[
= R_{\text{if } E \text{ then } \beta \text{ else } \gamma}
\]

\[
= R_{\alpha}
\]

Lemma 4.2.2 There is a formula of PDL that cannot be expressed in SPDL.

Proof: This is proved by Halpern and Reif in [5]. The idea of the proof is as follows: associated with every model is an edge-labelled directed graph, where the vertex set is the set of states, the edge set is the set of transitions of atomic programs, and edges are labelled with the names of the corresponding atomic programs. A tree model is a model for which the associated graph is a tree such that the edge sets corresponding to distinct atomic programs are disjoint. It is shown in [5] that every satisfiable formula is true at the root node of a tree model, and further, that every satisfiable formula of SPDL is true at the root node of a tree model with a polynomial number of nodes at each level. We now consider the PDL-formula \( F = [(\sigma \cup \tau)^*](\langle \sigma \rangle \text{true} \land \langle \tau \rangle \text{true}) \). This formula is satisfiable, and an example of a model for \( F \) is an infinite binary tree that can be drawn so that \( \sigma \) is the set of transitions from each node to its left hand child, and \( \tau \) is the set of transitions from each node to its right hand child. However it can be shown using induction that every tree model for \( F \) has at least \( 2^n \) nodes at level \( n \). Hence \( F \) cannot be expressed in SPDL. \( \square \)
**CHAPTER 4. EXPRESSIVE POWER**

**Theorem 4.2.3** PDL has greater expressive power than SPDL.

**Proof:** This follows from Lemmas 4.2.1 and 4.2.2. □

### 4.3 Extensions of SPDL

We now consider the extensions of SPDL by "∪" and "*", the logics SPDL+∪ and SPDL+∗. We are interested in whether or not the expressive powers of SPDL+∪ and SPDL+∗ are equivalent to the expressive power of PDL, or the expressive power of SPDL. We will show that SPDL+∗ and PDL have equivalent expressive power, and that PDL has greater expressive power than SPDL+∪, which has greater expressive power than SPDL.

We first show that each of SPDL+∪ and SPDL+∗ has greater expressive power than SPDL. As every formula of SPDL is also a formula of SPDL+∪ and SPDL+∗, it follows that every formula of SPDL can be expressed in both SPDL+∪ and SPDL+∗. We now show that there are formulas of SPDL+∪ and SPDL+∗ which cannot be expressed in SPDL.

**Lemma 4.3.1** There are formulas \( G \) of SPDL+∪ and \( H \) of SPDL+∗ which cannot be expressed in SPDL.

**Proof:** We use the result of Lemma 4.2.2 that the formula \([(σ ∪ τ)^*](⟨σ⟩\text{true} ∧ ⟨τ⟩\text{true})\) of PDL cannot be expressed in SPDL. We now consider the formulas 

\( G = [\text{while } (⟨σ⟩\text{true} ∧ ⟨τ⟩\text{true}) \text{ do } (σ ∪ τ)]\text{false} \) and \( H = [(σ^*; τ^*)^*](⟨σ⟩\text{true} ∧ ⟨τ⟩\text{true}) \).

We show that \( G \) and \( H \) are each equivalent to \( F = [(σ ∪ τ)^*](⟨σ⟩\text{true} ∧ ⟨τ⟩\text{true}) \), and hence cannot be expressed in SPDL. \( F \) is satisfied if \((⟨σ⟩\text{true} ∧ ⟨τ⟩\text{true})\) is true after the program \( (σ ∪ τ) \) has been repeated any finite number of times. \( G \) is satisfied if we can never reach a state in which \((⟨σ⟩\text{true} ∧ ⟨τ⟩\text{true})\) is false by repeating the program \( (σ ∪ τ) \). Hence \( F \) and \( G \) are equivalent. To show that \( F \) and \( H \) are equivalent notice that the programs \((σ ∪ τ)^* \) and \((σ^*; τ^*)^* \) are equivalent. □

Thus both SPDL+∪ and SPDL+∗ have greater expressive power than SPDL. We are now interested in whether either of SPDL+∪ and SPDL+∗ have expressive power equivalent
to PDL. It follows from Lemma 4.2.1 that every formula and program of SPDL+∪ and
SPDL+* can be expressed in PDL. To show that every formula of PDL can be expressed in
SPDL+* we need the following:

Definition 4.3.2 A program \( \alpha \) of PDL is in \( ∪ \)-normal form if \( \alpha = \beta_1 ∪ \ldots ∪ \beta_n \), where each
\( \beta_j \) is constructed from atomic programs and tests using only the operators ";" and "∗".

Lemma 4.3.3 Every program \( \alpha \) of PDL is equivalent to a program \( \alpha_∪ \) in \( ∪ \)-normal form.

Proof: By induction on the construction of programs. Let \( \alpha \) be a program of PDL. We
consider the possible cases for \( \alpha \):

Case 1: \( \alpha \in \Pi \).

Case 2: \( \alpha = F? \) for some \( F \in F_{\text{PDL}} \).

Case 3: \( \alpha = α^1; α^2 \). By induction there exist \( α_∪^1 = \beta_1^1 ∪ \ldots ∪ \beta_n^1 \), as above, for
\( i = 1, 2 \). Then

\[
\alpha = α^1; α^2
\]
\[
≡ α_∪^1; α_∪^2
\]
\[
≡ (β_1^1 ∪ \ldots ∪ β_n^1); (β_1^2 ∪ \ldots ∪ β_n^2)
\]
\[
≡ (β_1^1; β_1^2) ∪ (β_1^3; β_1^4) ∪ \ldots ∪ (β_1^i; β_1^j).
\]

Hence \( α_∪ = (β_1^1; β_1^2) ∪ (β_1^3; β_1^4) ∪ \ldots ∪ (β_1^i; β_1^j) \).

Case 4: \( \alpha = α^1 ∪ α^2 \).

Case 5: \( \alpha = δ^∗ \). By induction there exists \( δ_∪ = β_1 ∪ \ldots ∪ β_n \) as above. We use
the result that \( (α ∪ β)^∗ ≜ (α^∗; β^∗)^∗ \).

\[
\alpha = δ^∗
\]
\[
≡ δ_∪^∗
\]
\[
≡ (β_1 ∪ \ldots ∪ β_n)^∗
\]
\[
≡ (β_1^1; (β_2 ∪ \ldots ∪ β_n)^∗)^∗
\]
\[
≡ (β_1^1; (β_2^∗; (β_3 ∪ \ldots ∪ β_n)^∗)^∗)^∗
\]
\[
≡ (β_1^1; (β_2^∗; (β_3^∗; \ldots)^∗)^∗)^∗.
\]
Hence \( \alpha_\cup = (\beta_1^*; (\beta_2^*; (\ldots \beta_n^*)^*)^*). \)

\[ \square \]

We can now show that every formula of PDL can be expressed in SPDL+*:

**Lemma 4.3.4** Every formula of PDL can be expressed in SPDL+*.

**Proof:** We use induction on the construction of formulas and programs. The only difficult case is for formulas of the form \( F = [\alpha]G \). By induction we can assume that there is a formula \( G' \) of SPDL + * which is equivalent to \( G \). By Lemma 4.3.3 we know that every program of PDL can be expressed in \( \cup \)-normal form. Hence we can assume \( \alpha = \beta_1 \cup \ldots \cup \beta_n \) where each \( \beta_j \), \( 1 \leq j \leq n \), is constructed from atomic programs and tests using only the operators ; and *. By induction we can assume that for every test \( E? \) occurring in \( \beta_j \) there is a formula \( E' \) of SPDL + * which is equivalent to \( E \). Let \( \beta'_j \) be obtained from \( \beta_j \) by replacing each test \( E? \) by (if \( E' \) then skip else abort). Then each \( \beta'_j \) is a program of SPDL + *, and \( \alpha \equiv \beta'_1 \cup \ldots \cup \beta'_n \). Hence

\[
F = [\alpha]G
\equiv [\beta'_1 \cup \ldots \cup \beta'_n]G'
\equiv [\beta'_1]G' \land \ldots \land [\beta'_n]G'.
\]

Thus \( F' = [\beta'_1]G' \land \ldots \land [\beta'_n]G' \) is a formula of SPDL+* which is equivalent to \( F \). \( \square \)

We now have the following:

**Theorem 4.3.5** SPDL+* and PDL have equivalent expressive power.

We have shown that every formula and program of SPDL+* can be expressed in PDL, and every formula of PDL can be expressed in SPDL+*. However not every program of PDL can be expressed in SPDL+*. Consider the PDL-program \( \sigma \cup \tau \), and the model \( \mathcal{M}^\mathcal{U} = (S, \{ R_{\alpha} : \alpha \in \Pi \}, V) \), where \( S = \{ s, t, u \} \), \( R_{\sigma} = \{ (s, t) \} \), \( R_{\tau} = \{ (s, u) \} \), \( R_{\alpha} = \emptyset \) otherwise, and \( V(p) = S \) for every \( p \in \Phi \) (see Figure 4.1).
CHAPTER 4. EXPRESSIVE POWER

Figure 4.1: The model $\mathcal{M}^U$

From state $s$ the program $\sigma \cup \tau$ has transitions to both $t$ and $u$, but as there are no transitions from either $t$ or $u$ except to themselves any program of SPDL++ has a transition from $s$ to at most one of $t$ and $u$. Thus $\sigma \cup \tau$ is not equivalent to any program of SPDL++.

We know that every formula and program of SPDL+U can be expressed in PDL. We now show that there is a formula of PDL that cannot be expressed in SPDL+U, and hence that PDL has greater expressive power than SPDL+U. To do this we construct a class of infinite models $\{\mathcal{M}_d : d \in \mathbb{N}\}$ such that for each formula $F$ of SPDL+U either $V(F)$ or $V(\neg F)$ is finite in some model in the class, and show that there is a formula $G$ of PDL with both $V(G)$ and $V(\neg G)$ infinite in every model in the class.

Figure 4.2: The segment model $S_d$

For each $d \in \mathbb{N}$ we define a segment model $S_d$, where the transitions for the atomic programs $\sigma, \tau \in \Pi$ are as shown in Figure 4.2, and for every other atomic program $\pi \in \Pi$ we have $R_\pi = \emptyset$. Also $V(p) = \{s_0, \ldots, s_{d+1}\}$ for every $p \in \Phi$.

$\mathcal{M}_d$ is constructed by concatenating infinitely many copies of $S_d$, identifying endpoints of adjacent copies, and adjoining an additional state $\bar{s}$, as shown in Figure 4.3. Let $\bar{s} \in V(p)$ for every $p \in \Phi$. 
CHAPTER 4. EXPRESSION POWER

Figure 4.3: The model $\mathcal{M}_d$

Hence $\mathcal{M}_d = (S, \{R_\pi : \pi \in \Pi\}, V)$ where $S = \{\bar{s}, s_0^0, s_1^0, \ldots, s_{d+1}^0 = s_1^1, s_2^1, \ldots\}$, $R_\sigma$ and $R_\tau$ are as shown, $R_\pi = \emptyset$ for $\pi \neq \sigma, \tau$ and $V(p) = S$ for every $p \in \Phi$.

Lemma 4.3.6 There is a formula $G$ of PDL for which both $V(G)$ and $\overline{V(G)}$ are infinite in $\mathcal{M}_d$ for every $d \in N$.

Proof: Consider the PDL-formula $((\sigma^*; \tau; \sigma^*; \tau)^*)[\sigma \cup \tau]\text{false}$. We show this formula is true in states $s_0^t$ when $t$ is odd and false in states $s_0^t$ when $t$ is even. We observe that $\mathcal{M}_d, s \models [\sigma \cup \tau]\text{false}$ if and only if $s = \bar{s}$, hence $\mathcal{M}_d, s_0^t \models ((\sigma^*; \tau; \sigma^*; \tau)^*)[\sigma \cup \tau]\text{false}$ if and only if $s_0^t R_{(\sigma; \tau; \sigma; \tau)\bar{s}}$. From any state $s_h^t$ with $h \leq d$, the program $\sigma^*; \tau$ is a transition to either $s_{d-1}^t$ or $s_{d+1}^t = s_0^{t+1}$. From $s_{d-1}^t$ the program $\sigma^*; \tau$ is a transition to either $s_{d-2}^t$ or $s_0^t$, and from $s_0^{t+1}$ the program $\sigma^*; \tau$ is a transition to either $s_d^t$ or $s_0^{t+2}$. Hence from state $s_h^t$ with $h \leq d$ the program $\sigma^*; \tau; \sigma^*; \tau$ is a transition to one of $s_{d-2}^t$, $s_0^t$, $s_d^t$, or $s_0^{t+2}$. Notice that $sR_{(\sigma^*; \tau; \sigma^*; \tau)\bar{s}}$ if and only if $s = s_h^1$ for some $h \leq d$. Hence $s_0^t R_{(\sigma^*; \tau; \sigma^*; \tau)\bar{s}}$ if and only if $t \equiv 1 \mod 2$ i.e. if and only if $t$ is odd. Hence the formula is true in infinitely many states and false in infinitely many states.

Hence we have a formula of PDL which is true at infinitely many states and false at infinitely many states. To show that no formula of SPDL+U has this property we need the following:

Definition 4.3.7 An SPDL+U program $\alpha$ is in normal form if $\alpha = \beta_1 \cup \ldots \cup \beta_n$ where $\beta_j = \gamma_{j1}; \ldots; \gamma_{jm_j}$, for each $j, 1 \leq j \leq n$, and for each $\gamma_{jk}, 1 \leq k \leq m_j$, one of the following holds:

i) $\gamma_{jk} \in \Pi$

ii) $\gamma_{jk}$ is a test, i.e. of the form if $E$ then skip else abort, abbreviated $E$?

iii) $\gamma_{jk} = \text{while } E_{jk} \text{ do } \delta_{jk}$. 
**Lemma 4.3.8** Every program $\alpha$ of $\text{SPDL+U}$ is equivalent to a program $\alpha_N$ which is in normal form.

**Proof:** Use induction on programs.

Case 1: $\alpha \in \Pi$.

Case 2: $\alpha = \text{skip}$. Then $\alpha_N = \text{true}$?

Case 3: $\alpha = \text{abort}$. Then $\alpha_N = \text{false}$?

Case 4: $\alpha = \alpha^1; \alpha^2$. Suppose $\alpha^i$ has normal form $\alpha^i_N = \beta^i_1 \cup \ldots \cup \beta^i_n$. Then

\[
\alpha = \alpha^1; \alpha^2 \\
\equiv \alpha^1_N; \alpha^2_N \\
= (\beta^1_1 \cup \ldots \cup \beta^1_n_1); (\beta^2_1 \cup \ldots \cup \beta^2_n_2) \\
\equiv (\beta^1_1; \beta^2_1) \cup (\beta^1_2; \beta^2_2) \cup \ldots \cup (\beta^1_n_1; \beta^2_n_2).
\]

Hence $\alpha_N = (\beta^1_1; \beta^2_1) \cup (\beta^1_2; \beta^2_2) \cup \ldots \cup (\beta^1_n_1; \beta^2_n_2)$.

Case 5: $\alpha = \text{while } E \text{ do } \delta$.

Case 6: $\alpha = \text{if } E \text{ then } \delta^1 \text{ else } \delta^2$. Suppose $\delta^i$ has normal form $\delta^i_N = \beta^i_1 \cup \ldots \cup \beta^i_n$. Then

\[
\alpha = \text{if } E \text{ then } \delta^1 \text{ else } \delta^2 \\
\equiv (E?; \delta^1) \cup (\neg E?; \delta^2) \\
\equiv (E?; \delta^1_N) \cup (\neg E?; \delta^2_N) \\
\equiv (E?; (\beta^1_1 \cup \ldots \cup \beta^1_n)) \cup (\neg E?; (\beta^2_1 \cup \ldots \cup \beta^2_n)) \\
\equiv ((E?; \beta^1_1) \cup (E?; \beta^1_2) \cup \ldots \cup (E?; \beta^1_n)) \cup ((\neg E?; \beta^2_1) \cup \ldots \cup (\neg E?; \beta^2_n)) \\
\equiv (E?; \beta^1_1) \cup (E?; \beta^1_2) \cup \ldots \cup (E?; \beta^1_n) \cup (\neg E?; \beta^2_1) \cup \ldots \cup (\neg E?; \beta^2_n).
\]

Hence $\alpha_N = (E?; \beta^1_1) \cup (E?; \beta^1_2) \cup \ldots \cup (E?; \beta^1_n) \cup (\neg E?; \beta^2_1) \cup \ldots \cup (\neg E?; \beta^2_n)$.
Case 7: $a = a^1 \cup a^2$.

In each $M_d$ let $R_{E,\emptyset}(s)$ denote the set $\{ t \in S_d : s \text{ while } E \text{ do } \alpha t \}$. Intuitively this is the set of states reached from state $s$ in $M_d$ by the program "while $E$ do $\alpha$". For each $\beta \in \mathcal{P}$ let $R_{\beta}(s) = \{ t : s \beta t \}$.

We say a formula $F$ is finite (cofinite) in $M_d$ if $\bar{V}(F)$ is finite (cofinite) in $M_d$. In the next lemma we prove that for any formula $F$ of $SPDL + U$ we can find $d$ such that $F$ is either finite or cofinite in $M_d$. We in fact show that $F$ is either finite or cofinite in $M_d$ for all sufficiently large $d$. To prove this we also show that for every program of the form "while $E$ do $\alpha$" the sets $R_{E,\emptyset}(s)$ satisfy certain conditions for all sufficiently large $d$.

The conditions Wilm uses in [10] are very similar. As well as showing that every formula $F$ is either finite or cofinite in $M_d$ whenever $d$ is sufficiently large, Wilm shows there exists $t \in N$ such that $F$ is either true in every state above $s^t_0$ in $M_d$ or false in every state above $s^t_0$ in $M_d$ whenever $d$ is sufficiently large. For the assertion about while-programs we will show that for all but finitely many states the sets $R_{E,\emptyset}(s)$ satisfy certain conditions whenever $d$ is sufficiently large. Wilm shows that there exists $t \in N$ such that the conditions are satisfied for every state above $s^t_0$ in $M_d$ whenever $d$ is sufficiently large. We now state and prove the lemma.

**Lemma 4.3.9** (i) For every formula $F$ of $SPDL + U$ there exists $d_F \in N$ such that $F$ is either finite or cofinite in $M_d$ for all $d \geq d_F$.

(ii) For every program of the form "while $E$ do $\alpha$" of $SPDL + U$ there exists $d_{E,\emptyset} \in N$ such that for all $d \geq d_{E,\emptyset}$ one of the following holds for $M_d$:

1. (P)(i): $\{ s : R_{E,\emptyset}(s) \neq \{ s \} \}$ is finite.
2. (P)(ii): there is a finite set $S_{E,\emptyset}$ of states of $M_d$ such that for all but finitely many $s \in S$, $R_{E,\emptyset}(s) = S_{E,\emptyset}$.

**Proof:** We use induction on the construction of formulas and programs to prove (i) and (ii) simultaneously. We first consider the statement about formulas. There are four possible cases:
Case 1: $F = p$ for some $p \in \Phi$. By the definition of $V$ in $M_d$ we have $V(F) = S$, hence $F$ is cofinite in $M_d$ for $d \geq d_F = 0$.

Case 2: $F = \neg G$. By induction there exist $d_G \in N$ such that $G$ is either finite or cofinite in $M_d$ for all $d \geq d_G$. Set $d_F = d_G$, and let $d \geq d_F$. $F$ is finite in $M_d$ if $G$ is cofinite, and cofinite if $G$ is finite.

Case 3: $F = G \lor H$. Set $d_F = \max\{d_G, d_H\}$ and let $d \geq d_F$. If both $G$ and $H$ are finite in $M_d$ then so is $F$, otherwise at least one of $G$ and $H$ is cofinite in $M_d$, and hence so is $F$.

Case 4: $F = [\alpha]G$. We show that for every formula $E$ of SPDL+$\lor$ if there exists $d_E$ such that $E$ is either finite or cofinite in $M_d$ for every $d \geq d_E$, then for every subprogram $\alpha'$ of $\alpha$ there also exists $d_{[\alpha']E}$ such that $[\alpha']E$ is either finite or cofinite in $M_d$ for every $d \geq d_{[\alpha']E}$. The proof is by induction on the construction of programs.

Case 4.1: $\alpha' \in \Pi$, so $\alpha'$ is an atomic program. If $\alpha' \neq \sigma, \tau$ then $R_{\alpha'} = \emptyset$ and $\models [\alpha']E$ for every formula $E$, and $[\alpha']E$ is cofinite in $M_d$ for $d \geq 0$. For $\alpha' = \sigma, \tau$ note that for any state $s$ both $\sigma$ and $\tau$ are transitions to a state at most $d+1$ states away. Hence if $E$ is finite (cofinite) in $M_d$ then so are $[\sigma]E$ and $[\tau]E$. Hence $[\alpha']E$ is finite or cofinite in $M_d$ for $d \geq d_{[\alpha']E} = d_E$.

Case 4.2: $\alpha' = \text{skip}$. Suppose $E$ is finite or cofinite in $M_d$ for every $d \geq d_E$. We know $\models [\text{skip}]E \leftrightarrow E$ for any formula $E$ from the axiom schema DUM. Set $d_{[\alpha']E} = d_E$ and let $d \geq d_{[\alpha']E}$. Then $[\alpha']E$ is finite (cofinite) in $M_d$ if $E$ is.

Case 4.3 $\alpha' = \text{abort}$. Since $R_{\text{abort}} = \emptyset$ we have $\models [\text{abort}]E$ for any formula $E$, hence $[\alpha']E$ is cofinite in $M_d$ for $d \geq d_{[\alpha']E} = 0$.

Case 4.4 $\alpha' = \beta; \gamma$. We know $\models [\beta; \gamma]E \leftrightarrow [\beta][\gamma]E$ for any formula $E$ by the axiom schema COMP. Suppose $E$ is finite or cofinite in $M_d$ for every $d \geq d_E$. From the induction hypothesis we know for any formula $E'$ if there exists $d_{E'}$ such that $E'$ is either finite or cofinite in $M_d$ for $d \geq d_{E'}$ then there exist $d_{[\beta]E'}, d_{[\gamma]E'} \in N$ such that $[\beta]E'$ is either finite or cofinite in $M_d$ for $d \geq d_{[\beta]E'}$, and $[\gamma]E'$ is either finite or cofinite in $M_d$ for $d \geq d_{[\gamma]E'}$. For $E' = E$ we get $d_{[\gamma]E} \in N$ such that $[\gamma]E$ is either finite or cofinite
in $\mathcal{M}_d$ for $d \geq d_{[\gamma]}E$. Then for $E' = [\gamma]E$ we get $d_{[\beta][\gamma]}E \in N$ such that $[\beta][\gamma]E$ is either finite or cofinite in $\mathcal{M}_d$ for $d \geq d_{[\beta][\gamma]}E$. Hence $[\beta; \gamma]E$ is either finite or cofinite in $\mathcal{M}_d$ for $d \geq d_{[\beta;\gamma]E} = d_{[\beta][\gamma]}E$.

**Case 4.5 $\alpha' =$ while $H$ do $\beta$.** By induction there exist $d_{H,\beta} \in N$ such that for all $d \geq d_{H,\beta}$ either $(P)(i)$ or $(P)(ii)$ holds for $\mathcal{M}_d$. Suppose $E$ is finite or cofinite in $\mathcal{M}_d$ for $d \geq d_E$. Set $d_{[\alpha']E} = \max\{d_{H,\beta}, d_E\}$. Then for $d \geq d_{[\alpha']E}$ one of the following occurs:

**Case 4.5.1:** $(P)(i)$ holds. Then $\{s : RH,\beta(s) \neq R_{\text{skip}}(s)\}$ is finite, and there is $t_{H,\beta} \in N$ such that $RH,\beta(s^t_h) = R_{\text{skip}}(s^t_h) = \{s^t_h\}$ for every $t \geq t_{H,\beta}$ and $h \leq d + 1$. Note that $\{s^t_h : t < t_{H,\beta}\}$ is finite. If $E$ is finite (cofinite) in $\mathcal{M}_d$ then $\{s^t_h : t \geq t_{H,\beta}, h \leq d + 1, \mathcal{M}_d, s^t_h \models E\}$ is finite (cofinite). It follows that $\{s^t_h : t \geq t_{H,\beta}, h \leq d + 1, \mathcal{M}_d, s^t_h \models [\text{while } H \text{ do } \beta]E\}$ is finite (cofinite) and so $[\alpha']E$ is finite (cofinite) in $\mathcal{M}_d$.

**Case 4.5.2:** $(P)(ii)$ holds. Then there is a finite set $S_{H,\beta}$ of states of $\mathcal{M}_d$ such that for all but finitely many $s \in S$, $RH,\beta(s) = S_{H,\beta}$. Hence there is $t_{H,\beta} \in N$ such that $RH,\beta(s^t_h) = S_{H,\beta}$ for every $t \geq t_{H,\beta}$ and $h \leq d + 1$. Again note that $\{s^t_h : t < t_{H,\beta}\}$ is finite. If $\mathcal{M}, s \models E$ for every $s \in S(H,\beta)$ then $\mathcal{M}, s^t_h \models [\text{while } E \text{ do } \beta]E$ for every state $s^t_h$ such that $t \geq t_{H,\beta}$ and $h \leq d + 1$, and $[\alpha']E$ is cofinite in $\mathcal{M}_d$. Otherwise there is a state $s \in S(H,\beta)$ such that $\mathcal{M}, s \not\models E$, and in this case $\mathcal{M}, s^t_h \models \neg[\text{while } E \text{ do } \beta]E$ for every state $s^t_h$ such that $t \geq t_{H,\beta}$ and $h \leq d + 1$, and $[\alpha']E$ is finite in $\mathcal{M}_d$.

**Case 4.6: $\alpha' =$ if $H$ then $\beta$ else $\gamma$.** From the axiom schema COND we know $\models [\text{if } H \text{ then } \beta \text{ else } \gamma]E \leftrightarrow ((H \rightarrow [\beta]E) \land (\neg H \rightarrow [\gamma]E))$ for any formula $E$. Suppose $E$ is finite in $\mathcal{M}_d$ for sufficiently large $d$. By the induction hypothesis for Case 4, $[\beta]E$ and $[\gamma]E$ are also each either finite or cofinite in $\mathcal{M}_d$ for sufficiently large $d$. Also by the main induction hypothesis $H$ (and hence $\neg H$) is also either finite or cofinite in $\mathcal{M}_d$ for sufficiently large $d$. Set $d_{[\alpha']E} = \max\{d_{[\beta]E}, d_{[\gamma]E}, d_H\}$. Then for $d \geq d_{[\alpha']E}$, $[\text{if } H \text{ then } \beta \text{ else } \gamma]E$ is either finite or cofinite in $\mathcal{M}_d$.

**Case 4.7: $\alpha' = \beta \cup \gamma$.** We know $[\beta \cup \gamma]E \leftrightarrow ([\beta]E \land [\gamma]E)$ for every formula $E$ by the axiom schema ALT. Suppose $E$ is finite or cofinite in $\mathcal{M}_d$ for sufficiently large $d$. By the induction hypothesis $[\beta]E$ and $[\gamma]E$ are also each either finite or cofinite in $\mathcal{M}_d$ for
sufficiently large $d$, hence so is $([\beta]E \land [\gamma]E) = [\alpha']E$.

We now prove the statement about while-programs. Given a program of the form $\textbf{while} \ E \ \textbf{do} \ \alpha$, we consider the normal form of the program $\alpha$, $\alpha_N = \beta_1 \cup \ldots \cup \beta_n$ where $\beta_j = \gamma_{j1}; \ldots; \gamma_{jm}$, for each $j$ ($1 \leq j \leq n$), and each $\gamma_{ji}$ is either an atomic program, a test, or a program of the form $\textbf{while} \ F \ \textbf{do} \ \delta$.

For each $j$ ($1 \leq j \leq n$) we define the following:

$$T_j = \{F : \text{either } F? = \gamma_{ji} \text{ or } \neg F? = \gamma_{ji} \text{ for some } i, 1 \leq i \leq m_j\}$$

$$d_{T_j} = \max\{d_F : F \in T_j\}.$$ 

We can now define

$$T = \bigcup_{j \leq n} T_j$$

$$d_T = \max\{d_{T_j} : 1 \leq j \leq n\}.$$ 

If none of the $\gamma_{ji}$'s are tests then set $d_T = 0$.

If $\gamma_{ji}$ is the program $\textbf{while} \ F \ \textbf{do} \ \delta$ then by induction there exists $d_{F,\delta} \in N$ such that for all $d \geq d_{F,\delta}$ either (P)(i) or (P)(ii) holds in $M_d$. If some $\gamma_{ji}$ is a while-program then we can define the following

$$W_j = \{\textbf{while} \ F \ \textbf{do} \ \delta : \gamma_{ji} = \textbf{while} \ F \ \textbf{do} \ \delta \text{ for some } i, 1 \leq i \leq m_j\}$$

$$d_{W_j} = \max\{d_{F,\delta} : \textbf{while} \ F \ \textbf{do} \ \delta \in W_j\}.$$ 

We can now define

$$W = \bigcup_{j \leq n} W_j$$

$$d_W = \max\{d_{W_j} : 1 \leq j \leq n\}.$$ 

Otherwise if no $\gamma_{ji}$ is a while-program then set $d_W = 0$.

Each $\beta_j$ is a sequence of atomic programs, tests and while-programs. As $R_{\pi} = \emptyset$ for every $\pi \notin \{\sigma, \tau\}$, each $\beta_j$ with non-aborting executions is a sequence of occurrences of $\sigma$, $\tau$, tests and while-programs. Let $f_j$ be the greatest number of times $\sigma$ occurs consecutively
in \( \beta_j \), interspersed with tests and \texttt{while}-programs but not with any occurrences of \( \tau \). Let 
\[ f_{\omega} = \max\{ f_j : 1 \leq j \leq n \} \]
By the induction hypothesis for \( E \) there exists \( d_E \in \mathbb{N} \) such that \( E \) is either finite or cofinite in \( \mathcal{M}_d \) for every \( d \geq d_E \). Set 
\[ d_{E,\omega} = \max\{ d_E, d_T, d_W, 2f_{\omega} + 1 \} \]

Let \( d \geq d_{E,\omega} \) be fixed. Then each formula \( F \in T \) is either finite or cofinite in \( \mathcal{M}_d \), \( E \) is also either finite or cofinite in \( \mathcal{M}_d \), and either \( P(i) \) or \( P(ii) \) holds for \( \mathcal{M}_d \) for every \texttt{while}-program in \( W \). Let \( \hat{t} \) be such that \( E \), and every \( F \in T \), are either true in every state \( s_h^t \) with \( t \geq \hat{t} \), or false in every state \( s_h^t \) with \( t \geq \hat{t} \), and for every \texttt{while}-program in \( W \) either \( \text{R}_{F,\delta}(s_h^t) = \{ s_h^t \} \) for every state \( s_h^t \) with \( t \geq \hat{t} \), or \( \text{R}_{F,\delta}(s_h^t) = S_{F,\delta} \) for every state \( s_h^t \) with \( t \geq \hat{t} \). Set \( t_{E,\omega} = \hat{t} + 1 \). Then for every \( d \geq d_{E,\omega} \), each test occurring in one of the \( \beta_j \)'s is either equivalent to \texttt{skip} in every state \( s_h^t \) with \( t \geq t_{E,\omega} - 1 \), and \( h \leq d + 1 \), or equivalent to \texttt{abort} in every state \( s_h^t \) with \( t \geq t_{E,\omega} - 1 \), and \( h \leq d + 1 \). Every \texttt{while}-program is either equivalent to \texttt{skip} in every state \( s_h^t \) with \( t \geq t_{E,\omega} - 1 \), and \( h \leq d + 1 \), or a transition to a state below \( s_0^{t_{E,\omega}} \) in every state \( s_h^t \) with \( t \geq t_{E,\omega} - 1 \), and \( h \leq d + 1 \). Hence for each \( \beta_j \) which has non-aborting executions in state \( s_0^{t_{E,\omega}} \) or some state above \( s_0^{t_{E,\omega}} \) there is a sequence of occurrences of \( \sigma, \tau, \texttt{skip} \) and \texttt{while}-programs for which \( (P)(ii) \) holds, such that any execution of \( \beta_j \) in or above state \( s_0^{t_{E,\omega}} \) is equivalent to an execution of the concatenation of this sequence until a state below \( s_0^{t_{E,\omega}} \) is reached.

We need to show that \( (P)(i) \) or \( (P)(ii) \) holds for \texttt{while} \( E \) \texttt{do} \( \alpha \) in \( \mathcal{M}_d \). We consider two cases:

Case 1: \( E \) is finite in \( \mathcal{M}_d \). Then \( \mathcal{M}_d, s_h^t \models \neg E \) for every \( t \geq t_{E,\omega} \) and \( h \leq d + 1 \), so from the definition of \( \text{R}_{\text{while} E \text{ do} \alpha} \) we have \( \text{R}_{E,\omega}(s_h^t) = \{ s_h^t \} \) for every \( t \geq t_{E,\omega} \) and \( h \leq d + 1 \), and \( (P)(i) \) holds for \( \mathcal{M}_d \).

Case 2: \( E \) is cofinite in \( \mathcal{M}_d \). For every \( t \geq t_{E,\omega} - 1 \) and \( h \leq d + 1 \) we have \( \mathcal{M}, s_h^t \models E \), and each \( \beta_j \) is equivalent at \( s_h^t \) to a sequence of occurrences of \( \sigma, \tau \) and programs of the form \texttt{"while} \( F \) \texttt{do} \( \delta \)" as described above. Suppose for some fixed \( t' \geq t_{E,\omega} \) and \( h' \leq d + 1 \) that \( s_h^{t'} \text{R}_{\text{while} E \text{ do} \alpha} s' \) where \( s' \) is a state in \( \mathcal{M}_d \). We show that \( s_h^{t'} \text{R}_{\text{while} E \text{ do} \alpha} s' \) for every \( t \geq t_{E,\omega} \) and \( h \leq d + 1 \).
We consider the section of $\mathcal{M}_d$ between states $s_{d+1}^{t_{E,\alpha}}$ and $s_0^{t_{E,\alpha}}$, shown in Figure 4.4. We call this the buffer zone. As $s_h^{t_{E,\alpha}},\text{while } E \text{ do } a,s'$ there is a sequence $\beta_{j1}, \ldots, \beta_{jk}$ such that $s_h^{t_{E,\alpha}},R_{\beta_{j1}}; \ldots; R_{\beta_{jk}}s'$, and we know $\mathcal{M},s' \models E$, so $s'$ must be below state $s_0^{t_{E,\alpha}}$. For each program "while $F$ do $\delta$" which occurs in one of the $\beta_j$'s either condition (P)(i) or (P)(ii) holds for $\mathcal{M}_d$ for every state $s_h^t$ with $t \geq t_{E,\alpha} - 1$ and $h \leq d + 1$. We consider 2 cases:

Case 2.1: no program of the form "while $F$ do $\delta$" for which (P)(ii) holds occurs in any of $\beta_{j1}, \ldots, \beta_{jk}$, or the system is at a state below $s_0^{t_{E,\alpha}}$ when the first such program occurs.

By the choice of $d_{E,\alpha}$ and $t_{E,\alpha}$, for each state in the buffer zone the execution of each $\beta_j$ is equivalent (until a state below $s_0^{t_{E,\alpha}}$ is reached) to a sequence of occurrences of $\sigma$ and $\tau$ as described above, and $\mathcal{M}_d,s_h^{t_{E,\alpha}} \models E$ for every $h \leq d + 1$. Also $s_h^{t_{E,\alpha}}$ is above the buffer zone and $s'$ is below it, so the program $\beta_{j1}; \ldots; \beta_{jk}$ has transitions through this section. Further, it follows from the structure of $\mathcal{M}_d$ that every state between $s_h^{t_{E,\alpha}}$ and $s'$ (which includes every state in the buffer zone) must be visited.

We make the following observations:

Observation 1: For every $t \geq t_{E,\alpha} - 1$ no program $\beta_j$ is a transition from a state above $s_d^t$ to a state below $s_0^t$.

Justification: To reach a state below $s_0^t$ from a state above $s_d^t$ we must have a program with a sequence of $d$ consecutive occurrences of $\sigma$. As we chose $d_{E,\alpha} \geq 2f_\sigma + 1$ none of the programs $\beta_j$, $1 \leq j \leq n$, satisfy this condition.

Observation 2: there is a program $\beta_k$ which is equivalent to a sequence of occurrences of $\sigma$ when executed in any state $s_h^t$ with $t \geq t_{E,\alpha} - 1$ and $h \leq d + 1$ (provided
no state below $s_0^{t_{E, \alpha} - 1}$ is reached).

Justification: Suppose not. Every occurrence of $\tau$ at a state $s_h^{t_{E, \alpha} - 1}$ for $0 < h < d$ forces a transition back to $s_{d+1}^{t_{E, \alpha} - 1}$, and as we chose $d_{E, \alpha} \geq 2f_\sigma + 1$ if every $\beta_j$ contains an occurrence of $\tau$ then we can never reach state $s_0^{t_{E, \alpha} - 1}$ (or below) from a state above $s_d^{t_{E, \alpha} - 1}$. This is a contradiction, as the program $\beta_j; \ldots; \beta_{jk}$ reaches a state below $s_0^{t_{E, \alpha} - 1}$ from a state above $s_d^{t_{E, \alpha} - 1}$.

Observation 3: There is a program $\beta_i$ with an odd number of occurrences of $\tau$.

Justification: Consider a program $\beta_i'$ in the sequence $\beta_j; \ldots; \beta_{jk}$ such that the system is in or above state $s_{d+1}^{t_{E, \alpha} - 1}$ before executing $\beta_i'$ and in a state below $s_d^{t_{E, \alpha} - 1}$ after executing $\beta_i'$. The first occurrence of $\tau$ in this program must occur at state $s_{d+1}^{t_{E, \alpha} - 1}$, as if it occurred above $s_{d+1}^{t_{E, \alpha} - 1}$ then there would be a transition to $s_d^{t_{E, \alpha} - 1}$ and, by Observation 1, $\beta_i'$ couldn’t reach a state below $s_0^{t_{E, \alpha} = s_{d+1}^{t_{E, \alpha} - 1}}$. So the first occurrence of $\tau$ is a transition from $s_{d+1}^{t_{E, \alpha} - 1}$ to $s_d^{t_{E, \alpha} - 1}$. Now suppose there is a second occurrence of $\tau$. This must cause a transition back to $s_{d+1}^{t_{E, \alpha} - 1}$ (by the choice of $d_{E, \alpha} \geq 2f_\sigma + 1$) and the system will stay in this state until a third occurrence of $\tau$, which must happen as $\beta_i'$ ends in a state below $s_{d+1}^{t_{E, \alpha} - 1}$. Clearly $\beta_i'$ must contain an odd number of occurrences of $\tau$. Set $\beta_i = \beta_i'$.

Observation 4: There must be a state $s_{h'}^{t_{E, \alpha} - 1}$, $0 < h'' < d$ such that

$s_{h'}^{t_{E, \alpha} - 1} \overline{\text{while } E \text{ do } \alpha s'}$.

Justification: The final state reached by the program $\beta_i$ is below $s_{d+1}^{t_{E, \alpha} - 1}$, and must be above $s_0^{t_{E, \alpha} - 1}$ by Observation 1. Let $s_{h''}^{t_{E, \alpha} - 1}$ denote this state. Clearly

$s_{h''}^{t_{E, \alpha} - 1} \overline{\text{while } E \text{ do } \alpha s'}$.

Observation 5: For every $t \geq t_{E, \alpha}$ and $h \leq d+1$ there is a program $\alpha_{t, h}$ which is a concatenation of a finite sequence of programs $\beta_j$ ($1 \leq j \leq n$) such that $s_t^{t_{E, \alpha} - 1} R_{\alpha_{t, h}}^{t_{E, \alpha} - 1}$.

Justification: Set $n = t - t_{E, \alpha} + 1$. Define $\alpha_{t, h} = ((\beta_k)^d; \beta_l)^n$.

Intuitively the program $\alpha_{t, h}$ is the following: repeat $\beta_k$ enough (at most $d$) times to reach state $s_t^i$, then use $\beta_l$ to reach a state $s_h^{t_i - 1}$. Repeat this procedure to reach $s_h^{t_i - 2}$, then $s_h^{t_i - 3}$ and so on until state $s_h^{t_{E, \alpha}}$ is reached. Repeating $\beta_k$ another $d$ times we reach $s_h^{t_{E, \alpha}}$, then $\beta_l$ will reach $s_h^{t_{E, \alpha} - 1}$.

Hence for every $t \geq t_{E, \alpha}$ and $h \leq d+1$ there is program $\alpha_{t, h}$ such that

$s_t^{t_{E, \alpha} - 1} R_{\alpha_{t, h}}^{t_{E, \alpha} - 1} \overline{\text{while } E \text{ do } \alpha s'}$ and as $\alpha_{t, h}$ is a the concatenation of a finite sequence of
programs $\beta_j$ (1 $\leq$ $j$ $\leq$ $n$) it follows that $s^t_h R_{\text{while } E \text{ do } s'}$, as required.

Thus there is a set of states $S(E, \alpha)$ such that for every $t \geq t_{E, \alpha}$ and $h \leq d + 1$, $R_{E, \alpha}(s^t_h) = S(E, \alpha)$ in $M_d$. As each state $s$ in $S(E, \alpha)$ is below $s^t_{0E, \alpha}$, and there are only finitely many states below $s^t_{0E, \alpha}$, then $S(E, \alpha)$ must be finite and (P)(ii) holds for $M_d$.

Case 2.2: there is a while-program in $\beta_{j1}; \ldots; \beta_{jk}$ for which (P)(ii) holds and which is first reached at a state above $s^t_{0E, \alpha}$. Let while $F$ do $\delta$ be the first such program to occur, and suppose it occurs in $\beta_i$, when the system is in state $s''_{t''}$ for some $t'' \geq t_{E, \alpha} - 1$ and $h'' \leq d + 1$. Let $k$ be such that $\gamma_{ik}$ is the first occurrence of while $F$ do $\delta$ in $\beta_i$. Set $\gamma' = \gamma_{i1}; \ldots; \gamma_{i,k-1}$ and $\gamma'' = \gamma_{i,k+1}; \ldots; \gamma_{im}$, so $\beta_i = \gamma'; \gamma_{ik}; \gamma''$. By induction there is a finite set of states $S(F, \delta)$ such that $S^t_H(F, \delta) = S(F, \delta)$ for every $t \geq t_{E, \alpha} - 1$ and $h \leq d + 1$. In particular $S^t_{H''}(F, \delta) = S(F, \delta)$. As $s^t_H R_{\text{while } E \text{ do } s'}$, we must have $s R_{\text{while } E \text{ do } s'}$ for some $s \in S(F, \delta)$, and there is a state $s''$ such that $s R_{s''} s'' R_{\text{while } E \text{ do } s'}$. Hence

We now consider a general state $s^t_h$ in $M_d$, where $t \geq t_{E, \alpha}$ and $h \leq d + 1$. At this state $\gamma'$ is equivalent to a sequence of occurrences of $\sigma$ and $\tau$, and can only reach states above $s^t_{0E, \alpha}$. (as $s^t_h$ is above $s^t_{dE, \alpha}$ and, by Observation 1 proved for the base case, none of the $\beta_j$'s can be a transition from a state above $s^t_{dE, \alpha}$ to a state below $s^t_{0E, \alpha}$). So there is a state $s''_{t''}$ such that $s^t_h R_{\gamma'} s''_{t''}$ and $s''_{t''}$ is above $s^t_{0E, \alpha}$, which means $S^t_{H''}(F, \delta) = S(F, \delta)$. In particular $s''_{t''} R_{\text{while } E \text{ do } s^t}$, where $s$ is as above, so $s R_{s''} s'' R_{\text{while } E \text{ do } s'}$. Hence

Lemma 4.3.10 There is a formula of PDL which cannot be expressed in $SPDL + \cup$.

Proof: It follows from Lemma 4.3.9 that for every formula $F$ of $SPDL + \cup$ there is a $d$ such that either $V(F)$ or $V(\neg F)$ is finite in $M_d$. We showed in Lemma 4.3.6 that there is a
formula $G$ of PDL such that both $\overline{V}(G)$ and $\overline{V}(\neg G)$ are infinite in $M_d$ for every $d \in N$, hence $G$ is a formula of PDL which cannot be expressed in SPDL+$\cup$. \hfill \Box$

We have now proved the following theorem:

**Theorem 4.3.11** *PDL has greater expressive power than SPDL+$\cup$*

Hence adding the non-deterministic operator $\cup$ to SPDL does not give expressive power equivalent to that of PDL. In order to get expressive power equivalent to PDL we must add the highly non-deterministic $*$ operator.
Bibliography


