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

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

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

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 "$\cup$" 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 $[\alpha]$, where $[\alpha]F$ is read "the formula F holds after every terminating execution of the program α". We define the dual operator $\langle\alpha\rangle$ by $\langle\alpha\rangle =_{def} \neg[\alpha]\neg$. The formula $\langle\alpha\rangle F$ 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 $\langle\alpha\rangle\mathbf{true}$ for "α terminates", and use formulas such as $F \rightarrow [\alpha]G$ 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 constucted as for PDL. We also consider two extensions of SPDL. The first is the extension of SPDL by the operator $\cup$ (SPDL+$\cup$), 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-deteŕministic operators "$\cup$" and "$*$". We show that both of these extensions have greater expresive 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+$\cup$ has less expressive power than PDL. We give an exposition of a proof by Wilm [10] that the extension of SPDL by $\cup$ 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+$\cup$ 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 $\cup$ 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_α and R_β be binary relations over the set S. We define the binary relation $R_\alpha \circ R_\beta$ by

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

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

$$\begin{aligned} R^0 &= id_S \\ R^{n+1} &= R^n \circ R. \end{aligned}$$

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 sR^*t if and only if sR^nt 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 $\neg, \vee,$ **true**, **false**, $;, \cup, *, ?, [,], ($, and $)$.

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

F1: $\Phi \subseteq \mathcal{F}$

F2: $\mathbf{true} \in \mathcal{F}$

F3: $\mathbf{false} \in \mathcal{F}$

F4: If $F \in \mathcal{F}$ then $\neg F \in \mathcal{F}$

F5: If $F, G \in \mathcal{F}$ then $(F \vee G) \in \mathcal{F}$

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

P1: $\Pi \subseteq \mathcal{P}$

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

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

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

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

$\mathcal{F}_{\mathrm{PDL}}$ is called the set of *formulas* of PDL and $\mathcal{P}_{\mathrm{PDL}}$ the set of *programs* of PDL.

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

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

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

(iii) the symbols $\neg$, $\vee$, **true**, **false**, ; , **skip**, **abort**, **while**, **do**, **if**, **then**, **else**, [,], (, and).

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

P1′: $\Pi \subseteq \mathcal{P}$

P2′: If $\alpha, \beta \in \mathcal{P}$ then $(\alpha; \beta) \in \mathcal{P}$

P3′: **skip** $\in \mathcal{P}$

P4′: **abort** $\in \mathcal{P}$

P5′: If $\alpha \in \mathcal{P}$ and $F \in \mathcal{F}$, then (**while** F **do** α) $\in \mathcal{P}$

P6′: If $\alpha, \beta \in \mathcal{P}$ and $F \in \mathcal{F}$, then (**if** F **then** α **else** β) $\in \mathcal{P}$.

$\mathcal{F}_{\text{SPDL}}$ is called the set of formulas of SPDL and $\mathcal{P}_{\text{SPDL}}$ the set of programs of SPDL.

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

The connectives are read as follows: ";" is concatenation so $\alpha; \beta$ means "do α then do β", "$\cup$" is disjunction so $\alpha \cup \beta$ means "do α or β" ($\cup$ is nondeterministic), α^* means "repeat α 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 α". The program **skip** is "do nothing", and **abort** means "abort".

We define $\wedge$, $\rightarrow$, and $\leftrightarrow$ as usual. The modal operator $\langle\ \rangle$, which is the dual of $[\ ]$, is defined by $\langle\alpha\rangle F =_{df} \neg[\alpha]\neg F$ and is read "there is a terminating execution of α after which F holds".

Axiom schemas for PDL:

PC: All tautologies

K: $[\alpha](F \to G) \to ([\alpha]F \to [\alpha]G)$

COMP: $[\alpha;\beta]F \leftrightarrow [\alpha][\beta]F$

ALT: $[\alpha \cup \beta]F \leftrightarrow [\alpha]F \wedge [\beta]F$

MIX: $[\alpha^*]F \to F \wedge [\alpha][\alpha^*]F$

IND: $[\alpha^*](F \to [\alpha]F) \to (F \to [\alpha^*]F)$

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

Rules of Inference for PDL:

MODUS PONENS: From F and $(F \to G)$ infer G

NECESSITATION: From F infer $[\alpha]F$.

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

Axiom schemas for SPDL:

PC: All tautologies

K: $[\alpha](F \to G) \to ([\alpha]F \to [\alpha]G)$

DUM: $[\mathbf{skip}]F \leftrightarrow F$

ABORT: $[\mathbf{abort}]\mathbf{false}$

COMP: $[\alpha;\beta]F \leftrightarrow [\alpha][\beta]F$

COND: $[\mathbf{if}\ E\ \mathbf{then}\ \alpha\ \mathbf{else}\ \beta]F \leftrightarrow ((E \to [\alpha]F) \wedge (\neg E \to [\beta]F)$

WHILE1: $\neg E \to ([\mathbf{while}\ E\ \mathbf{do}\ \alpha]F \to F)$

WHILE2: $[\mathbf{while}\ E\ \mathbf{do}\ \alpha]F \to (E \to [\alpha][\mathbf{while}\ E\ \mathbf{do}\ \alpha]F)$.

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 $(E \wedge F) \rightarrow [\alpha]F$ infer $F \rightarrow [\textbf{while } E \textbf{ do } \alpha](\neg E \wedge F)$.

As above, let Th(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 Th(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_α is a binary relation on S for each $\alpha \in \mathcal{P}$, and V is a function from Φ to $\wp(S)$. The members of S are called *states*. Intuitively the pairs in R_α are the possible *transitions* of the program α, 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:

S1: $\mathcal{M}, s \models p$ if $s \in V(p)$, for $p \in \Phi$

S2: $\mathcal{M}, s \models \textbf{true}$

S3: $\mathcal{M}, s \not\models \textbf{false}$

S4: $\mathcal{M}, s \models \neg F$ if $\mathcal{M}, s \not\models F$

S5: $\mathcal{M}, s \models F \vee G$ if $\mathcal{M}, s \models F$ or $\mathcal{M}, s \models G$

S6: $\mathcal{M}, s \models [\alpha]F$ if for all $s' \in S$, $sR_\alpha s'$ implies $\mathcal{M}, s' \models F$.

From the definitions of the symbols $\wedge, \rightarrow, \leftrightarrow$ and of the modal operator $\langle\ \rangle$ we obtain the following equivalences:

(i) $\mathcal{M}, s \models F \wedge G$ 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 $sR_\alpha s'$ and $\mathcal{M}, s' \models F$.

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

S1′: $V \subseteq \overline{V}$

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

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

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

S5′: $\overline{V}(F \vee G) = \overline{V}(F) \cup \overline{V}(G)$.

S6′: $\overline{V}([\alpha]F) = \{s \in S : (\forall s' \in S)(sR_\alpha s' \Rightarrow s' \in \overline{V}(F))$.

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 $\wedge, \rightarrow, \leftrightarrow$, and the modal operator $\langle\ \rangle$ we obtain the following equivalences:

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

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

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

$$\overline{V}(\langle\alpha\rangle F) = \{s \in S : (\exists s' \in S)(sR_\alpha s' \text{ and } s' \in \overline{V}(F)).$$

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:

M1: $R_{\alpha;\beta} = R_\alpha \circ R_\beta$

M2: $R_{\alpha\cup\beta} = R_\alpha \cup R_\beta$

M3: $R_{\alpha^*} = (R_\alpha)^*$

M4: $R_{E?} = \{(s,s) : \mathcal{M}, s \models E\}$.

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:

M2′: $R_{\mathbf{skip}} = id_S$

M3′: $R_{\mathbf{abort}} = \emptyset$

M4′: $R_{\mathbf{while}\ E\ \mathbf{do}\ \alpha} = \{(s,s') : (\exists n)(\exists s_0, \ldots s_n)(s = s_0,\ s' = s_n,\ \mathcal{M}, s_n \models \neg E$ and $(\forall j < n)((s_j, s_{j+1}) \in R_\alpha$ and $\mathcal{M}, s_j \models E))\}$

M5′: $R_{\mathbf{if}\ E\ \mathbf{then}\ \alpha\ \mathbf{else}\ \beta} = \{(s,s') : sR_\alpha s'$ and $\mathcal{M}, s \models E\} \cup \{(s,s') : sR_\beta s'$ 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 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 Λ be a set of formulas of a logic $\mathcal{L}$. Λ is *consistent* if $\neg(F_1 \wedge \ldots \wedge F_n)$ 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 α^n for each $n \in N$ as follows:

$$\begin{aligned} \alpha^0 &= \textbf{skip} \\ \alpha^{n+1} &= \alpha^n ; \alpha. \end{aligned}$$

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 $\alpha \in \mathcal{P}$:

$\mathrm{Sf}(p) = \{p\}$
$\mathrm{Sf}(\neg F) = \{\neg F\} \cup \mathrm{Sf}(F)$
$\mathrm{Sf}(F \vee G) = \{F \vee G\} \cup \mathrm{Sf}(F) \cup \mathrm{Sf}(G)$
$\mathrm{Sf}([\alpha]F) = \{[\alpha]F\} \cup \mathrm{Sf}(F)$.

Note that formulas occurring in α 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:

$$\begin{aligned}
\mathcal{M}, s \models [\alpha;\beta]F &\overset{\text{S6}}{\iff} (\forall t)(sR_{\alpha;\beta}t \Rightarrow \mathcal{M}, t \models F) \\
&\overset{\text{M1}}{\iff} (\forall t)(sR_{\alpha} \circ R_{\beta}t \Rightarrow \mathcal{M}, t \models F) \\
&\iff (\forall t)(\forall u)(sR_{\alpha}uR_{\beta}t \Rightarrow \mathcal{M}, t \models F) \\
&\iff (\forall u)(sR_{\alpha}u \Rightarrow ((\forall t)(uR_{\beta}t \Rightarrow \mathcal{M}, t \models F))) \\
&\overset{\text{S6}}{\iff} (\forall u)(sR_{\alpha}u \Rightarrow \mathcal{M}, u \models [\beta]F) \\
&\overset{\text{S6}}{\iff} \mathcal{M}, s \models [\alpha][\beta]F.
\end{aligned}$$

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

ALT:

$$\begin{aligned}\mathcal{M},s\models[\alpha\cup\beta]F &\overset{S6}{\iff} \forall t\ (sR_{\alpha\cup\beta}t\Rightarrow\mathcal{M},t\models F)\\ &\overset{M2}{\iff} \forall t\ (sR_\alpha t\Rightarrow\mathcal{M},t\models F)\text{ and }\forall t(sR_\beta t\Rightarrow\mathcal{M},t\models F)\\ &\overset{S6}{\iff} \mathcal{M},s\models[\alpha]F\text{ and }\mathcal{M},s\models[\beta]F\\ &\iff \mathcal{M},s\models[\alpha]F\wedge[\beta]F.\end{aligned}$$

Hence $\mathcal{M},s\models[\alpha\cup\beta]F\leftrightarrow([\alpha]F\wedge[\beta]F)$.

MIX: Suppose $\mathcal{M},s\models[\alpha^*]F$, and let t be such that $sR_\alpha t$. Then

$$tR_{\alpha^*}u\overset{M3}{\Longrightarrow}t(R_\alpha)^*u\Longrightarrow s(R_\alpha)^*u\Longrightarrow sR_{\alpha^*}u\Longrightarrow\mathcal{M},u\models F.$$

Thus for all t, if $sR_\alpha t$ then $\mathcal{M},t\models[\alpha^*]F$, so $\mathcal{M},s\models[\alpha][\alpha^*]F$. Also by definition $sR_{\alpha^*}s$, hence $\mathcal{M},s\models F$. Thus $\mathcal{M},s\models[\alpha^*]F\rightarrow F\wedge[\alpha][\alpha^*]F$.

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

TEST:

$$\begin{aligned}\mathcal{M},s\models[F?]G &\overset{S6}{\iff} \forall t(sR_{F?}t\Rightarrow\mathcal{M},t\models G)\\ &\overset{M4}{\iff} \forall t(s=t\text{ and }\mathcal{M},s\models F)\Rightarrow\mathcal{M},t\models G\\ &\iff \mathcal{M},s\models F\Rightarrow\mathcal{M},s\models G\\ &\iff \mathcal{M},s\models(F\rightarrow G).\end{aligned}$$

Hence $\mathcal{M},s\models[F?]G\leftrightarrow(F\rightarrow G)$.

□

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 $\mathcal{M}^C = \langle S^C, \{R^C_\alpha : \alpha \in \mathcal{P}_{\mathrm{PDL}}\}, V^C\rangle$ where

- S^C is the set of all maximal PDL-consistent sets of formulas
- for each $\alpha \in \mathcal{P}_{\mathrm{PDL}}$, R^C_α is the binary relation defined by
  $sR^C_\alpha t \iff \{G \in \mathcal{F} : [\alpha]G \in s\} \subseteq t$
- $V^C : \Phi \to \wp(S^C)$ 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* $s \in S^C$, $\mathcal{M}^C, s \models F \iff F \in s$.

Proof: By induction on formulas. The details can be found in [8] pp 23–25 or in [4] p25. ☐

It follows that every theorem of PDL is valid in $\mathcal{M}^C$, and no non-theorem of PDL is valid in $\mathcal{M}^C$. However $\mathcal{M}^C$ 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 Γ of PDL formulas is called *FL-closed* if it is closed under subformulas and the following conditions hold for all $F, G \in \mathcal{F}_{\mathrm{PDL}}$ and $\alpha, \beta \in \mathcal{P}_{\mathrm{PDL}}$:

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

$$s \sim t \iff (\forall F \in \Gamma)(\mathcal{M}, s \models F \iff \mathcal{M}, t \models F).$$

We introduce the following notations:

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

Let $V^\Gamma : \Phi^\Gamma \to \wp(S^\Gamma)$ be defined by the equivalence

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

Definition 3.3.3 For $\alpha \in \mathcal{P}_{\mathrm{PDL}}$, a binary relation R^Γ on S^Γ is a (Γ, α)-*filtration* of the binary relation R_α 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 (Γ, α)-filtration of R_α is the smallest filtration,

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

The model $\mathcal{M}^\Gamma = \langle S^\Gamma, \{R_\alpha^\Gamma : \alpha \in \mathcal{P}^\Gamma\}, V^\Gamma\rangle$ is a Γ-*filtration* of $\mathcal{M} = \langle S, \{R_\alpha : \alpha \in \mathcal{P}\}, V\rangle$ if R_α^Γ is a (Γ, α)-filtration of R_α for every $\alpha \in \mathcal{P}^\Gamma$.

Lemma 3.3.4 *Let $\mathcal{M}^\Gamma = \langle S^\Gamma, \{R_\alpha^\Gamma : \alpha \in \mathcal{P}^\Gamma\}, V^\Gamma\rangle$ be a Γ-filtration of the model $\mathcal{M} = \langle S, \{R_\alpha : \alpha \in \mathcal{P}\}, V\rangle$. 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. ◫

Lemma 3.3.5 *Suppose that $\mathcal{M}^\Gamma = \langle S^\Gamma, \{R^\Gamma_\alpha : \alpha \in \mathcal{P}^\Gamma\}, V^\Gamma\rangle$ is a Γ-filtration of the model $\mathcal{M} = \langle S, \{R_\alpha : \alpha \in \mathcal{P}_{PDL}\}, V\rangle$ such that Γ is finite, and let T be a subset of S^Γ. Then there is a formula A_T such that for all $s \in S$, $\mathcal{M}, s \models A_T$ if and only if $[s] \in T$.*

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

$$\begin{aligned} \mathcal{M}, s \models A_t &\iff (\mathcal{M}, s \models F \Leftrightarrow \mathcal{M}, t \models F) \text{ for all } F \in \Gamma \\ &\iff s \sim t \\ &\iff [s] = [t]. \end{aligned}$$

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

$$\begin{aligned} \mathcal{M}, s \models A_T &\overset{S5}{\iff} \mathcal{M}, s \models A_{t_1} \text{ or} \ldots \text{or } \mathcal{M}, s \models A_{t_n} \\ &\iff [s] = [t_1] \text{ or} \ldots \text{or } [s] = [t_n] \\ &\iff [s] \in T. \end{aligned}$$

□

Let A be a non-theorem of PDL. Let Γ_A be the smallest FL-closed set containing A.

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

- R^Γ_π is a (Γ_A, π)-filtration of R^C_π for each $\pi \in \Pi^{\Gamma_A}$
- R^Γ_π is arbitrary for each $\pi \in \Pi \setminus \Pi^{\Gamma_A}$
- $R^\Gamma_{F?} = \{([s],[s]) : \mathcal{M}^C, s \models F\}$ for each $F? \in \mathcal{P}^\Gamma$
- R^Γ_α is defined by the standard model conditions for PDL otherwise.

We now show that Γ_A is finite, so we can apply Lemma 3.3.5.

Lemma 3.3.6 *Γ_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 occurences in α. We show that in the construction of Γ_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):

(ii) Every subformula of $[\alpha][\beta]F$, except for $[\alpha][\beta]F$ and $[\beta]F$, is also a subformula of $[\alpha;\beta]F$,

(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 $\Gamma_A = \cup_{n\leq m} A_n$ is finite. □

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

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

Case 1: $\alpha \in \Pi^\Gamma$. Then R^Γ_α is a Γ_A-filtration of R^C_α by definition of R^Γ_α.

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

$$\{G \in \mathcal{F} : [F?]G \in s\} \subseteq t \tag{3.1}$$

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

$$\begin{aligned} G \in s \quad &\Longrightarrow \quad F \to G \in s \\ &\overset{\text{TEST}}{\Longleftrightarrow} \quad [F?]G \in s \\ &\overset{(3.1)}{\Longrightarrow} \quad G \in t. \end{aligned}$$

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

$$\begin{aligned} (F \to F) \in s \quad &\overset{\text{TEST}}{\Longleftrightarrow} \quad [F?]F \in s \\ &\overset{(3.1)}{\Longrightarrow} \quad F \in t \\ &\Longleftrightarrow \quad F \in s \\ &\overset{3.3.1}{\Longleftrightarrow} \quad \mathcal{M}^C, s \models F. \end{aligned}$$

Hence

$$\begin{aligned} sR^C_{F?}t \quad &\Longrightarrow \quad (s = t \text{ and } \mathcal{M}^C, s \models F) \\ &\Longleftrightarrow \quad [s]R^\Gamma_{F?}[s]. \end{aligned}$$

so G1 holds.

G2: Suppose $[s]R^{\Gamma}_{F?}[t]$, and also that $[F?]G \in \Gamma_A$ and $\mathcal{M}^C, s \models [F?]G$. Then by definition of $R^{\Gamma}_{F?}$ we have $[t] = [s]$ and $\mathcal{M}^C, s \models F$. The axiom schema TEST yields $\mathcal{M}^C, s \models (F \to G)$, and hence $\mathcal{M}^C, s \models G$. But $G \in \Gamma_A$ (as Γ_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 $[s]R^{\Gamma}_{\beta;\gamma}[t]$ (such an A_s exists by Lemma 3.3.5). By the induction hypothesis G1 holds for R^C_β and R^C_γ. We now show that $\mathcal{M}^C, s \models [\beta][\gamma]A_s$:

$$\begin{aligned} sR^C_\beta u R^C_\gamma t &\implies [s]R^{\Gamma}_\beta[u]R^{\Gamma}_\gamma[t] \\ &\iff [s]R^{\Gamma}_{\beta;\gamma}[t] \\ &\iff A_s \in t \\ &\overset{3.3.1}{\iff} \mathcal{M}^C, t \models A_s. \end{aligned}$$

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

$$\begin{aligned} \mathcal{M}^C, s \models [\beta][\gamma]A_s &\overset{\text{Comp}}{\iff} \mathcal{M}^C, s \models [\beta;\gamma]A_s \\ &\overset{\text{S6}}{\iff} sR^C_{\beta;\gamma}t \Rightarrow \mathcal{M}^C, t \models A_s \\ &\overset{3.3.1}{\iff} sR^C_{\beta;\gamma}t \Rightarrow A_s \in t \\ &\iff sR^C_{\beta;\gamma}t \Rightarrow [s]R^{\Gamma}_{\beta;\gamma}[t]. \end{aligned}$$

Hence G1 holds.

G2: Suppose $[s]R^{\Gamma}_{\beta;\gamma}[t]$, 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:

$$[s]R^{\Gamma}_{\beta;\gamma}[t] \overset{\text{M1}}{\implies} \exists[u]([s]R^{\Gamma}_\beta[u]R^{\Gamma}_\gamma[t]) \tag{3.2}$$

$$[\beta;\gamma]G \in \Gamma_A \implies [\beta][\gamma]G \in \Gamma_A \text{ and } [\gamma]G \in \Gamma_A \tag{3.3}$$

$$\mathcal{M}^C, s \models [\beta;\gamma]G \overset{\text{COMP}}{\iff} \mathcal{M}^C, s \models [\beta][\gamma]G. \tag{3.4}$$

By the induction hypothesis G2 holds for R^C_β and R^C_γ. Using the above implications we now show that G2 holds for $R^C_{\beta;\gamma}$:

$$\begin{array}{cl}
 & [s]R^\Gamma_{\beta;\gamma}[t] \text{ and } [\beta;\gamma]G \in \Gamma_A \text{ and } \mathcal{M}^C, s \models [\beta;\gamma]G \\
\overset{(3.2)(3.3)(3.4)}{\Longrightarrow} & \exists[u]([s]R^\Gamma_\beta[u]R^\Gamma_\gamma[t] \text{ and } [\beta][\gamma]G \in \Gamma_A \text{ and } [\gamma]G \in \Gamma_A \text{ and } \mathcal{M}^C, s \models [\beta][\gamma]G) \\
\Longrightarrow & \exists[u]([u]R^\Gamma_\gamma[t] \text{ and } [\gamma]G \in \Gamma_A \text{and} \\
 & ([s]R^\Gamma_\beta[u] \text{and } [\beta][\gamma]G \in \Gamma_A \text{ and } \mathcal{M}^C, s \models [\beta][\gamma]G)) \\
\overset{\text{IH}}{\Longrightarrow} & \exists[u]([u]R^\Gamma_\gamma[t] \text{ and } [\gamma]G \in \Gamma_A \text{ and } \mathcal{M}^C, u \models [\gamma]G) \\
\overset{\text{IH}}{\Longrightarrow} & \mathcal{M}^C, t \models G
\end{array}$$

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 $[s]R^\Gamma_{\beta\cup\gamma}[t]$. By the induction hypothesis G1 holds for R^C_β and R^C_γ. We now show that $\mathcal{M}^C, s \models [\beta]A_s \wedge [\gamma]A_s$. First we need the following implication:

$$\begin{array}{rcl}
sR^C_\beta t & \Rightarrow & [s]R^\Gamma_\beta[t] \\
 & \Rightarrow & [s]R^\Gamma_{\beta\cup\gamma}[t] \\
 & \Rightarrow & A_s \in t \\
 & \overset{3.3.1}{\Rightarrow} & \mathcal{M}^C, t \models A_s.
\end{array}$$

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

$$\begin{array}{rcl}
\mathcal{M}^C, s \models ([\beta]A_s \wedge [\gamma]A_s) & \overset{\text{ALT}}{\Longleftrightarrow} & \mathcal{M}^C, s \models [\beta \cup \gamma]A_s \\
 & \overset{\text{S6}}{\Longleftrightarrow} & sR^C_{\beta\cup\gamma}t \Rightarrow \mathcal{M}^C, t \models A_s \\
 & \overset{3.3.1}{\Longleftrightarrow} & sR^C_{\beta\cup\gamma}t \Rightarrow A_s \in t \\
 & \Longleftrightarrow & sR^C_{\beta\cup\gamma}t \Rightarrow [s]R^\Gamma_{\beta\cup\gamma}[t].
\end{array}$$

Hence G1 holds.

G2: Suppose $[s]R^{\Gamma}_{\beta\cup\gamma}[t]$, 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:

$$\begin{aligned} [s]R^{\Gamma}_{\beta\cup\gamma}[t] &\overset{M2}{\Rightarrow} [s]R^{\Gamma}_{\beta}[t] \text{ or } [s]R^{\Gamma}_{\gamma}[t] && (3.5)\\ [\beta\cup\gamma]G \in \Gamma_A &\Rightarrow [\beta]G \in \Gamma_A \text{ and } [\gamma]G \in \Gamma_A && (3.6)\\ \mathcal{M}^C, s \models [\beta\cup\gamma]G &\overset{ALT}{\Rightarrow} \mathcal{M}^C, s \models ([\beta]G \wedge [\gamma]G) && \\ &\Rightarrow \mathcal{M}^C, s \models [\beta]G \text{ and } \mathcal{M}^C, s \models [\gamma]G. && (3.7) \end{aligned}$$

By the induction hypothesis G2 holds for R^C_β and R^C_γ. Using this, and the above implications, we have

$$\begin{aligned} & [s]R^{\Gamma}_{\beta\cup\gamma}[t] \text{ and } [\beta\cup\gamma]G \in \Gamma_A \text{ and } \mathcal{M}^C, s \models [\beta\cup\gamma]G \\ \overset{(3.5)(3.6)(3.7)}{\Longrightarrow}\ & ([s]R^{\Gamma}_{\beta}[t] \text{ or } [s]R^{\Gamma}_{\gamma}[t]) \text{ and } [\beta]G \in \Gamma_A \text{ and } [\gamma]G \in \Gamma_A \\ & \text{and } \mathcal{M}^C, s \models [\beta]G \text{ and } \mathcal{M}^C, s \models [\gamma]G \\ \Longrightarrow\ & ([s]R^{\Gamma}_{\beta}[t] \text{ and } [\beta]G \in \Gamma_A \text{ and } \mathcal{M}^C, s \models [\beta]G) \\ & \text{or } ([s]R^{\Gamma}_{\gamma}[t] \text{ and } [\gamma]G \in \Gamma_A \text{ and } \mathcal{M}^C, s \models [\gamma]G) \\ \overset{IH}{\Longrightarrow}\ & \mathcal{M}^C, t \models G. \end{aligned}$$

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 $[s]R^{\Gamma}_{\beta^*}[t]$. We now show that $\mathcal{M}^C, s \models (A_s \rightarrow [\beta^*]A_s)$. We need the following chain of implications:

$$\begin{aligned}
\mathcal{M}^C, t \models A_s &\overset{3.3.1}{\Longrightarrow} A_s \in t \\
&\Longrightarrow [s]R^{\Gamma}_{\beta^*}[t] \\
&\overset{M3}{\Longrightarrow} [s](R^{\Gamma}_{\beta})^*[t] \\
&\Longrightarrow \exists n([s](R^{\Gamma}_{\beta})^n[t]) \\
&\Longrightarrow \exists n([s](R^{\Gamma}_{\beta})^n[t]) \text{ and } (tR_\beta u \overset{IH}{\Rightarrow} [t]R^{\Gamma}_{\beta}[u]) \\
&\Longrightarrow tR_\beta u \Rightarrow (\exists n([s](R^{\Gamma}_{\beta})^n[t]) \text{ and } [t]R^{\Gamma}_{\beta}[u]) \\
&\Longrightarrow tR_\beta u \Rightarrow \exists n([s](R^{\Gamma}_{\beta})^{n+1}[u]) \\
&\Longrightarrow tR_\beta u \Rightarrow ([s](R^{\Gamma}_{\beta})^*[u]) \\
&\Longrightarrow tR_\beta u \Rightarrow [s]R^{\Gamma}_{\beta^*}[u] \\
&\Longrightarrow tR_\beta u \Rightarrow A_s \in u \\
&\overset{3.3.1}{\Longrightarrow} tR_\beta u \Rightarrow \mathcal{M}^C, u \models A_s \\
&\overset{S6}{\Longrightarrow} \mathcal{M}^C, t \models [\beta]A_s.
\end{aligned}$$

Hence $\mathcal{M}^C, t \models A_s \to [\beta]A_s$, and this is true for all t. In particular if $sR_{\beta^*}t$ then $\mathcal{M}^C, t \models A_s \to [\beta]A_s$, so $\mathcal{M}^C, s \models [\beta^*](A_s \to [\beta]A_s)$. The axiom schema IND yields $\mathcal{M}^C, s \models (A_s \to [\beta^*]A_s)$. As $[s](R^{\Gamma}_{\beta})^0[s]$ we know from the definition of $R^{\Gamma}_{\beta^*} = (R^{\Gamma}_{\beta})^*$ that $[s]R^{\Gamma}_{\beta^*}[s]$. From this we get the following chain of implications:

$$\begin{aligned}
[s]R^{\Gamma}_{\beta^*}[s] &\Longrightarrow A_s \in s \\
&\overset{3.3.1}{\Longrightarrow} \mathcal{M}^C, s \models A_s \\
&\Longrightarrow \mathcal{M}^C, s \models [\beta^*]A_s \\
&\overset{S6}{\Longrightarrow} sR_{\beta^*}t \Rightarrow \mathcal{M}^C, t \models A_s \\
&\overset{3.3.1}{\Longrightarrow} sR_{\beta^*}t \Rightarrow A_s \in t \\
&\Longrightarrow sR_{\beta^*}t \Rightarrow [s]R^{\Gamma}_{\beta^*}[t].
\end{aligned}$$

Hence G1 holds.

G2: Suppose that $[s]R^{\Gamma}_{\beta^*}[t]$, 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 $n \geq 0$

$$([s]R^{\Gamma}_{\beta^n}[t] \text{ and } [\beta^*]G \in \Gamma \text{ and } \mathcal{M}, s \models [\beta^*]G) \Rightarrow \mathcal{M}, t \models [\beta^*]G. \tag{3.8}$$

Case $n = 0$: Suppose $[s]R^{\Gamma}_{\beta^0}[t]$, $[\beta^*]G \in \Gamma$ and $\mathcal{M}, s \models [\beta^*]G$. Then $[s] = [t]$, so as $[\beta^*]G \in \Gamma$, $\mathcal{M}, s \models [\beta^*]G$ implies $\mathcal{M}, t \models [\beta^*]G$.

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

$$\begin{aligned}
& [s]R^{\Gamma}_{\beta^{k+1}}[t] \text{ and } [\beta^*]G \in \Gamma \text{ and } \mathcal{M}, s \models [\beta^*]G \\
\Longrightarrow\ & \exists u([s]R^{\Gamma}_{\beta^k}[u]R^{\Gamma}_{\beta}[t]) \text{ and } [\beta^*]G \in \Gamma \text{ and } \mathcal{M}, s \models [\beta^*]G \\
\Longrightarrow\ & \exists u([s]R^{\Gamma}_{\beta^k}[u] \text{ and } [\beta^*]G \in \Gamma \text{ and } \mathcal{M}, s \models [\beta^*]G \text{ and } [u]R^{\Gamma}_{\beta}[t]) \\
\overset{(3.8)}{\Longrightarrow}\ & \exists u(\mathcal{M}, u \models [\beta^*]G \text{ and } [\beta^*]G \in \Gamma \text{ and } [u]R^{\Gamma}_{\beta}[t]) \\
\overset{\text{MIX}}{\Longrightarrow}\ & \exists u([u]R^{\Gamma}_{\beta}[t] \text{ and } [\beta][\beta^*]G \in \Gamma \text{ and } \mathcal{M}, u \models [\beta][\beta^*]G) \\
\overset{\text{IH}}{\Longrightarrow}\ & \mathcal{M}, t \models [\beta^*]G.
\end{aligned}$$

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

$$\begin{aligned}
& [s]R^{\Gamma}_{\beta^*}[t] \text{ and } [\beta^*]G \in \Gamma \text{ and } \mathcal{M}, s \models [\beta^*]G \\
\Longrightarrow\ & \exists n([s]R^{\Gamma}_{\beta^n}[t] \text{and } [\beta^*]G \in \Gamma \text{ and } \mathcal{M}, s \models [\beta^*]G) \\
\Longrightarrow\ & \mathcal{M}, t \models [\beta^*]G \\
\overset{\text{MIX}}{\Longrightarrow}\ & \mathcal{M}, t \models G.
\end{aligned}$$

Hence G2 holds.

□

Lemma 3.3.8 *$\mathcal{M}^{\Gamma} = \langle S^{\Gamma}, \{R^{\Gamma}_{\alpha} : \alpha \in \mathcal{P}^{\Gamma}\}, V^{\Gamma}\rangle$ is a standard model for PDL.*

Proof: We need to show that M1–M4 are satisfied. M1–M3 must hold by definition of R^{Γ}_{α}. For M4: for $F? \in \mathcal{P}^{\Gamma}$, $R_{F?} = \{([s],[s]) : \mathcal{M}^C, s \models F\}$, 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 $R_{F?} = \{([s],[s]) : \mathcal{M}^{\Gamma}, [s] \models F\}$ and M4 holds for every $F? \in \mathcal{P}^{\Gamma}$. For $F? \notin \mathcal{P}^{\Gamma}$ M4 holds by definition of R^{Γ}_{α}. □

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 Γ_A denote the smallest FL-closed set containing A. We showed in Lemma 3.3.7 that the model $\mathcal{M}^\Gamma$ is a Γ_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 $\mathcal{M}^C, s \models \neg A$ so $\mathcal{M}^C, s \not\models A$. As $A \in \Gamma_A$, then by Lemma 3.3.4 $\mathcal{M}^\Gamma, [s] \not\models A$. By Lemma 3.3.8 $\mathcal{M}^\Gamma$ 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 $\mathcal{M}$ denotes a standard model for SPDL.

DUM:

$$\begin{aligned} \mathcal{M}, s \models [\textbf{skip}]F &\overset{\text{S6}}{\iff} \forall t(sR_{\textbf{skip}}t \Rightarrow \mathcal{M}, t \models F) \\ &\overset{\text{M1}'}{\iff} \forall t(s = t \Rightarrow \mathcal{M}, t \models F) \\ &\iff \mathcal{M}, s \models F. \end{aligned}$$

Hence $\mathcal{M}, s \models [\textbf{skip}]F \leftrightarrow F$.

ABORT: By condition M2′ we have $R_{\textbf{abort}} = \emptyset$. Let s be a state in a standard SPDL-model. For every state t, $sR_{\textbf{abort}}t$ implies $\mathcal{M}, t \models F$, hence $\mathcal{M}, s \models [\textbf{abort}]F$.

COND:

$$\begin{aligned}
& \mathcal{M}, s \models [\textbf{if } E \textbf{ then } \alpha \textbf{ else } \beta]F \\
\overset{\text{S6}}{\iff}\ & \forall t(sR_{\text{if } E \text{ then } \alpha \text{ else } \beta}t \Rightarrow \mathcal{M}, t \models F) \\
\overset{\text{M5}'}{\iff}\ & \forall t(((\mathcal{M}, s \models E \text{ and } sR_\alpha t) \text{ or } (\mathcal{M}, s \models \neg E \text{ and } sR_\beta t)) \Rightarrow \mathcal{M}, t \models F) \\
\iff\ & \forall t(((\mathcal{M}, s \models E \text{ and } sR_\alpha t) \Rightarrow \mathcal{M}, t \models F) \\
& \text{and } ((\mathcal{M}, s \models \neg E \text{ and } sR_\beta t) \Rightarrow \mathcal{M}, t \models F)) \\
\iff\ & \forall t((\mathcal{M}, s \models E \text{ and } sR_\alpha t) \Rightarrow \mathcal{M}, t \models F) \\
& \text{and } \forall t((\mathcal{M}, s \models \neg E \text{ and } sR_\beta t) \Rightarrow \mathcal{M}, t \models F) \\
\iff\ & \forall t(\mathcal{M}, s \models E \Rightarrow (sR_\alpha t \Rightarrow \mathcal{M}, t \models F)) \text{ and } \forall t(\mathcal{M}, s \models \neg E \Rightarrow (sR_\beta t \Rightarrow \mathcal{M}, t \models F)) \\
\iff\ & (\mathcal{M}, s \models E \Rightarrow \forall t(sR_\alpha t \Rightarrow \mathcal{M}, t \models F)) \text{ and } (\mathcal{M}, s \models \neg E \Rightarrow \forall t(sR_\beta t \Rightarrow \mathcal{M}, t \models F)) \\
\overset{\text{S6}}{\iff}\ & (\mathcal{M}, s \models E \Rightarrow \mathcal{M}, s \models [\alpha]F) \text{ and } (\mathcal{M}, s \models \neg E \Rightarrow \mathcal{M}, s \models [\beta]F) \\
\iff\ & (\mathcal{M}, s \models E \rightarrow [\alpha]F) \text{ and } (\mathcal{M}, s \models \neg E \rightarrow [\beta]F) \\
\iff\ & \mathcal{M}, s \models (E \rightarrow [\alpha]F) \wedge (\neg E \rightarrow [\beta]F)
\end{aligned}$$

Hence $\mathcal{M}, s \models [\textbf{if } E \textbf{ then } \alpha \textbf{ else } \beta]F \leftrightarrow ((E \rightarrow [\alpha]F) \wedge (\neg E \rightarrow [\beta]F))$.

WHILE1:

$$\begin{aligned}
\mathcal{M}, s \models \neg E \ \overset{\text{S6}}{\iff}\ & \forall t(sR_{\textbf{while } E \textbf{ do } \alpha}t \Rightarrow s = t) \\
\Longrightarrow\ & sR_{\textbf{while } E \textbf{ do } \alpha}s \\
\Longrightarrow\ & (\mathcal{M}, s \models [\textbf{while } E \textbf{ do } \alpha]F \Rightarrow \mathcal{M}, s \models F) \\
\Longrightarrow\ & \mathcal{M}, s \models [\textbf{while } E \textbf{ do } \alpha]F \rightarrow F.
\end{aligned}$$

Hence $\mathcal{M}, s \models \neg E \rightarrow [\textbf{while } E \textbf{ do } \alpha]F \rightarrow F$.

WHILE2: Suppose $\mathcal{M}, s \models [\textbf{while } E \textbf{ do } \alpha]F$ and $\mathcal{M}, s \models E$. We need to show that $\mathcal{M}, s \models [\alpha][\textbf{while } E \textbf{ do } \alpha]F$. Let t be such that $sR_\alpha t$. Then for every u such that $tR_{\textbf{while } E \textbf{ do } \alpha}u$ we have $sR_{\textbf{while } E \textbf{ do } \alpha}u$ so $\mathcal{M}, u \models F$, and hence $\mathcal{M}, t \models [\textbf{while } E \textbf{ do } \alpha]F$. Hence $\mathcal{M}, t \models [\textbf{while } E \textbf{ do } \alpha]F$ for every t such that $sR_\alpha t$, and so as required we get $\mathcal{M}, s \models [\alpha][\textbf{while } E \textbf{ do } \alpha]F$.

HOARE'S ITERATION RULE: Suppose $\mathcal{M} \models (E \wedge F) \rightarrow [\alpha]F$. Let s be any state such that $\mathcal{M}, s \models F$. We show that $\mathcal{M}, s \models$ [**while** E **do** α]F. Let t be such that $sR_{\text{while E do }\alpha}t$. Then there exist $n \in N$ 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 $\mathcal{M}, s_k \models (E \wedge F)$ 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 $sR_{\text{while E do }\alpha}t$ implies $\mathcal{M}, t \models F$, so $\mathcal{M}, s \models$ [**while** E **do** α]F. Thus $\mathcal{M}, s \models F \rightarrow$ [**while** E **do** α]F for every $s \in S$, so $\mathcal{M} \models F \rightarrow$ [**while** E **do** α]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 $\mathcal{M}^C = \langle S^C, \{R^C_\alpha : \alpha \in \mathcal{P}_{\text{SPDL}}\}, V^C \rangle$ where

- S^C is the set of all maximal SPDL-consistent sets
- For each $\alpha \in \mathcal{P}_{\text{SPDL}}$, R^C_α is the binary relation defined by
  $sR^C_\alpha t \iff \{G \in \mathcal{F} : [\alpha]G \in s\} \subseteq t$
- $V^C : \Phi \rightarrow \wp(S^C)$ 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 Γ of SPDL-formulas is called *FL_S-closed* if it is closed under subformulas and the following conditions hold for all $F, G \in \mathcal{F}_{\mathrm{SPDL}}$ and $\alpha, \beta \in \mathcal{P}_{\mathrm{SPDL}}$:

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

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

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

Let Γ be a $\mathrm{FL_S}$-closed set of SPDL formulas. As for PDL we define the notions of a (Γ, α)-filtration of a binary relation, and a Γ-filtration of a model.

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

Proof: As for PDL. □

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

Lemma 3.5.4 *Γ_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 $[\alpha][\textbf{while } E \textbf{ do } \alpha]F$, except $[\alpha][\textbf{while } E \textbf{ do } \alpha]F$ itself, is a subformula of $[\textbf{while } E \textbf{ do } \alpha]F$.

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

$\square$

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

- R^Γ_π is a Γ_A-filtration of R^C_π for $\pi \in \Pi^{\Gamma_A}$
- R^Γ_π is arbitrary for $\pi \in \Pi \setminus \Pi^{\Gamma_A}$
- $R^\Gamma_{\textbf{skip}} = id_{S^\Gamma}$
- $R^\Gamma_{\textbf{abort}} = \emptyset$
- R^Γ_α is defined by the standard model conditions for SPDL otherwise.

We will prove simultaneously by induction that for every formula $F \in \Gamma_A$, $\mathcal{M}^C, s \models F$ if and only if $\mathcal{M}^\Gamma, [s] \models F$, and that R^Γ_α is a (Γ, α)-filtration of R^C_α 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 **while** or **if** statements.

$\mathcal{F}_n$ is the set of formulas of Γ_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 "**while** E **do** β" or "**if** E **then** β **else** γ" has $E \in \mathcal{F}_n$ and $\beta, \gamma \in \mathcal{P}_n$.

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

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

Proof: It suffices to prove that for every $n \in N$ the following hold:

(i) for every $\alpha \in \mathcal{P}_n$, R^Γ_α is a (Γ, α)-filtration of R^C_α

(ii) for every $F \in \mathcal{F}_n$, $\mathcal{M}^C, s \models F$ if and only if $\mathcal{M}^\Gamma, [s] \models F$.

We use strong induction on n.

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

Case 1: $\alpha \in \Pi^{\Gamma}$. Then R^{Γ}_{α} is a Γ_A-filtration of R^{C}_{α} by definition of R^{Γ}_{α}.

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

G1: We need the following implication:

$$\begin{aligned} sR^{C}_{\mathbf{skip}}t &\iff \{F \in \mathcal{F}_{\text{SPDL}} : [\mathbf{skip}]F \in s\} \subseteq t \\ &\overset{\text{DUM}}{\iff} \{F \in \mathcal{F}_{\text{SPDL}} : F \in s\} \subseteq t \\ &\implies s \subseteq t. \end{aligned}$$

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

G2: This is proved as follows:

$$\begin{aligned} & [s]R^{\Gamma}_{\mathbf{skip}}[t] \text{ and } \mathcal{M}, s \models [\mathbf{skip}]F \text{ and } [\mathbf{skip}]F \in \Gamma_A \\ \implies\ & s \sim t \text{ and } \mathcal{M}, s \models [\mathbf{skip}]F \text{ and } [\mathbf{skip}]F \in \Gamma_A \\ \implies\ & \mathcal{M}, t \models [\mathbf{skip}]F \\ \overset{\text{DUM}}{\iff}\ & \mathcal{M}, t \models F. \end{aligned}$$

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

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

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

$$\begin{aligned} \mathcal{M}^C, s \models p &\overset{\text{S1}}{\iff} s \in V(p) \\ &\iff [s] \in V^{\Gamma}(p) \\ &\overset{\text{S1}}{\iff} \mathcal{M}^{\Gamma}, [s] \models p. \end{aligned}$$

Case 2: F = **true**. From S2 we have both $\mathcal{M}^C, s \models \mathbf{true}$ and $\mathcal{M}^\Gamma, [s] \models \mathbf{true}$ hence $\mathcal{M}^C, s \models \mathbf{true}$ if and only if $\mathcal{M}^\Gamma, [s] \models \mathbf{true}$.

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

Case 4: $F = \neg G$. Then

$$\begin{aligned} \mathcal{M}^C, s \models F &\overset{\text{S4}}{\Longleftrightarrow} \mathcal{M}^C, s \not\models G \\ &\overset{\text{IH}}{\Longleftrightarrow} \mathcal{M}^\Gamma, [s] \not\models G \\ &\overset{\text{S4}}{\Longleftrightarrow} \mathcal{M}^\Gamma, [s] \models F. \end{aligned}$$

Case 5: $F = G \vee H$. Then

$$\begin{aligned} \mathcal{M}^C, s \models F &\overset{\text{S5}}{\Longleftrightarrow} \mathcal{M}^C, s \models G \text{ or } \mathcal{M}^C, s \models H \\ &\overset{\text{IH}}{\Longleftrightarrow} \mathcal{M}^\Gamma, [s] \models G \text{ or } \mathcal{M}^\Gamma, [s] \models H \\ &\overset{\text{S5}}{\Longleftrightarrow} \mathcal{M}^\Gamma, [s] \models F. \end{aligned}$$

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

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

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

$$\begin{aligned} sR^C_\alpha t &\overset{\text{G1}}{\Longrightarrow} [s]R^\Gamma_\alpha[t] \\ &\overset{\text{S6}}{\Longrightarrow} \mathcal{M}^\Gamma, [t] \models G \\ &\overset{\text{IH}}{\Longleftrightarrow} \mathcal{M}^C, t \models G. \end{aligned}$$

Hence $\mathcal{M}^C, s \models [\alpha]G$.

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: $\alpha =$ **while** E **do** β, where $E \in \mathcal{F}_n$ and $\beta \in \bigcup_{i \leq n} \mathcal{P}_i$.

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 $[s](E \mid R^{\Gamma}_{\beta})^m[t]$, where $(E \mid R^{\Gamma}_{\beta}) = \{([s],[t]) : [s]R^{\Gamma}_{\beta}[t] \text{ and } \mathcal{M}^{\Gamma},[s] \models E\}$ (such an A_s exists by Lemma 3.5.3). We now show that $\mathcal{M}^C \models (E \wedge A_s) \rightarrow [\beta]A_s$. Let u be any state such that $\mathcal{M}^C, u \models E \wedge A_s$, and suppose $uR_{\beta}t$.

$$\begin{array}{cl}
 & \mathcal{M}^C, u \models (E \wedge A_s) \text{ and } uR_{\beta}t \\
\overset{\text{IH}}{\Longrightarrow} & \mathcal{M}^C, u \models A_s \text{ and } \mathcal{M}^C, u \models E \text{ and } [u]R^{\Gamma}_{\beta}[t] \\
\overset{3.5.1,\text{IH}}{\Longrightarrow} & A_s \in u \text{ and } \mathcal{M}^{\Gamma},[u] \models E \text{ and } [u]R^{\Gamma}_{\beta}[t] \\
\Longleftrightarrow & \exists m([s](E \mid R_{\beta})^m[u]) \text{ and } \mathcal{M}^{\Gamma},[u] \models E \text{ and } [u]R^{\Gamma}_{\beta}[t] \\
\Longleftrightarrow & (\exists m)([s](E \mid R_{\beta})^m[u]) \text{ and } [u](E \mid R^{\Gamma}_{\beta})[t] \\
\Longrightarrow & \exists m([s](E \mid R_{\beta})^{m+1}[t]) \\
\Longleftrightarrow & A_s \in t \\
\overset{3.5.1}{\Longleftrightarrow} & \mathcal{M}^C, t \models A_s.
\end{array}$$

Hence

$$\begin{array}{rcl}
\mathcal{M}^C, u \models (E \wedge A_s) & \Longrightarrow & uR_{\beta}t \Rightarrow \mathcal{M}^C, t \models A_s \\
 & \Longrightarrow & \mathcal{M}^C, u \models [\beta]A_s.
\end{array}$$

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

$$\begin{array}{rcl}
sR_{\textbf{while } E \textbf{ do } \beta}t & \Longrightarrow & \mathcal{M}^C, t \models A_s \wedge \neg E \\
 & \Longleftrightarrow & \mathcal{M}^C, t \models A_s \text{ and } \mathcal{M}^C, t \models \neg E \\
 & \overset{3.5.1}{\Longleftrightarrow} & A_s \in t \text{ and } \mathcal{M}^C, t \models \neg E \\
 & \Longleftrightarrow & \exists m([s](E \mid R^{\Gamma}_{\beta})^m[t]) \text{ and } \mathcal{M}^C, t \models \neg E \\
 & \overset{\text{IH}}{\Longleftrightarrow} & \exists m([s](E \mid R^{\Gamma}_{\beta})^m[t]) \text{ and } \mathcal{M}^{\Gamma}, [t] \models \neg E \\
 & \overset{\text{M4}'}{\Longleftrightarrow} & [s]R^{\Gamma}_{\textbf{while } E \textbf{ do } \beta}[t].
\end{array}$$

Hence G1 holds.

G2: Suppose $[s]R_{\textbf{while } E \textbf{ do } \beta}[t]$, $[\textbf{while } E \textbf{ do } \beta]G \in \Gamma_A$ and $\mathcal{M}^C, s \models [\textbf{while } E \textbf{ do } \beta]G$. Then, as $[s]R_{\textbf{while } E \textbf{ do } \beta}[t]$, there exist $m \in N$ 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 $\mathcal{M}^C, s_i \models [\textbf{while } E \textbf{ do } \beta]G$ for every $i \leq m$.

Base step: $i = 0$. This is true by hypothesis.

Induction step: Assume $\mathcal{M}^C, s_k \models [\textbf{while } E \textbf{ do } \beta]G$, 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 [WHILE2:] we have $\mathcal{M}^C, s_k \models [\beta][\textbf{while } E \textbf{ do } \beta]G$. As Γ_A is FL$_S$-closed, $[\beta][\textbf{while } E \textbf{ do } \beta]G \in \Gamma_A$. We also know $[s_k]R^\Gamma_\beta[s_{k+1}]$. As $\beta \in \mathcal{P}_n$ we know R^Γ_β is a (Γ, β)-filtration of R^C_β and G2 holds. This yields $\mathcal{M}^C, s_{k+1} \models [\textbf{while } E \textbf{ do } \beta]G$.

Hence $\mathcal{M}^C, s_m \models [\textbf{while } E \textbf{ do } \beta]G$. 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 WHILE1 yields $\mathcal{M}^C, s \models G$, which is what was required to show G2 holds.

Case 2: $\alpha = \textbf{if } E \textbf{ then } \beta \textbf{ else } \gamma$

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

$$\begin{aligned}
\mathcal{M}^C, s \models E &\overset{\text{IH}}{\Longrightarrow} \mathcal{M}^\Gamma, s \models E \\
&\overset{\text{IH}}{\Longrightarrow} sR^C_\beta t \Rightarrow ([s]R^\Gamma_\beta[t] \text{ and } \mathcal{M}^\Gamma, s \models E) \\
&\overset{\text{IH}}{\Longrightarrow} sR^C_\alpha t \Rightarrow [s]R^\Gamma_{\textbf{if } E \textbf{ then } \beta \textbf{ else } \gamma}[t] \\
&\Longrightarrow sR^C_\beta t \Rightarrow A_s \in t \\
&\overset{3.5.1}{\Longrightarrow} sR^C_\beta t \Rightarrow \mathcal{M}^C, t \models A_s \\
&\Longrightarrow \mathcal{M}^C, s \models [\beta]A_s.
\end{aligned}$$

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

$$\begin{aligned}
\mathcal{M}^C, s \models [\textbf{if } E \textbf{ then } \beta \textbf{ else } \gamma]A_s &\overset{\text{S6}}{\Longleftrightarrow} sR^C_{\textbf{if } E \textbf{ then } \beta \textbf{ else } \gamma}t \Rightarrow \mathcal{M}^C, t \models A_s \\
&\overset{3.5.1}{\Longleftrightarrow} sR^C_{\textbf{if } E \textbf{ then } \beta \textbf{ else } \gamma}t \Rightarrow A_s \in t \\
&\Longleftrightarrow sR^C_{\textbf{if } E \textbf{ then } \beta \textbf{ else } \gamma}t \Rightarrow [s]R^\Gamma_{\textbf{if } E \textbf{ then } \beta \textbf{ else } \gamma}[t].
\end{aligned}$$

Hence G1 holds.

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

$$\begin{aligned}
[s]R^{\Gamma}_{\textbf{if } E \textbf{ then } \beta \textbf{ else } \gamma}[t] &\overset{\text{M5}'}{\iff} (\mathcal{M}^{\Gamma}, [s] \models E \text{ and } [s]R^{\Gamma}_{\beta}[t]) \\
&\qquad \text{or } (\mathcal{M}^{\Gamma}, [s] \models \neg E \text{ and } [s]R^{\Gamma}_{\gamma}[t]) && (3.9) \\
[\textbf{if } E \textbf{ then } \beta \textbf{ else } \gamma]G \in \Gamma_A &\implies [\beta]G \in \Gamma_A \text{ and } [\gamma]G \in \Gamma_A && (3.10) \\
\mathcal{M}, s \models [\textbf{if } E \textbf{ then } \beta \textbf{ else } \gamma]G &\overset{\text{COND}}{\iff} \mathcal{M}^C, s \models (E \to [\beta]G) \wedge (\neg E \to [\gamma]G) \\
&\iff \mathcal{M}^C, s \models E \to [\beta]G \\
&\qquad \text{and } \mathcal{M}^C, s \models \neg E \to [\gamma]G. && (3.11)
\end{aligned}$$

We now use these implications to show that G2 holds:

$$\begin{aligned}
& [s]R^{\Gamma}_{\textbf{if } E \textbf{ then } \beta \textbf{ else } \gamma}[t] \text{ and } [\textbf{if } E \textbf{ then } \beta \textbf{ else } \gamma]G \in \Gamma_A \\
& \text{and } \mathcal{M}, s \models [\textbf{if } E \textbf{ then } \beta \textbf{ else } \gamma]G \\
\overset{(3.9)(3.10)(3.11)}{\implies} \quad & ((\mathcal{M}^{\Gamma}, [s] \models E \text{ and } [s]R^{\Gamma}_{\beta}[t]) \text{ or } (\mathcal{M}^{\Gamma}, [s] \models \neg E \text{ and } [s]R^{\Gamma}_{\gamma}[t])) \\
& \text{and } [\beta]G \in \Gamma_A \text{ and } [\gamma]G \in \Gamma_A \text{ and } \mathcal{M}^C, s \models E \to [\beta]G \\
& \text{and } \mathcal{M}^C, s \models \neg E \to [\gamma]G \\
\overset{\text{IH}}{\iff} \quad & ((\mathcal{M}^C, s \models E \text{ and } [s]R^{\Gamma}_{\beta}[t]) \text{ or } (\mathcal{M}^C, s \models \neg E \text{ and } [s]R^{\Gamma}_{\gamma}[t])) \text{ and } [\beta]G \in \Gamma_A \\
& \text{and } [\gamma]G \in \Gamma_A \text{ and } \mathcal{M}^C, s \models E \to [\beta]G \text{ and } \mathcal{M}^C, s \models \neg E \to [\gamma]G \\
\implies \quad & (\mathcal{M}^C, s \models E \text{ and } [s]R^{\Gamma}_{\beta}[t] \text{ and } [\beta]G \in \Gamma_A \text{ and } \mathcal{M}^C, s \models E \to [\beta]G) \\
& \text{or } (\mathcal{M}^C, s \models \neg E \text{ and } [s]R^{\Gamma}_{\gamma}[t] \text{ and } [\gamma]G \in \Gamma_A \text{ and } \mathcal{M}^C, s \models \neg E \to [\gamma]G) \\
\implies \quad & ([s]R^{\Gamma}_{\beta}[t] \text{ and } [\beta]G \in \Gamma_A \text{ and } \mathcal{M}^C, s \models [\beta]G) \\
& \text{or } ([s]R^{\Gamma}_{\gamma}[t] \text{ and } [\gamma]G \in \Gamma_A \text{ and } \mathcal{M}^C, s \models [\gamma]G) \\
\overset{\text{IH}}{\implies} \quad & \mathcal{M}^C, t \models G.
\end{aligned}$$

Hence G2 holds.

Hence (i) holds for $\mathcal{P}_{n+1}$. The proof that (ii) holds for $\mathcal{F}_{n+1}$ is the same as for the base case.
□

Lemma 3.5.6 *$\mathcal{M}^\Gamma = \langle S^\Gamma, \{R^\Gamma_\alpha : \alpha \in \mathcal{P}^\Gamma\}, V^\Gamma\rangle$ 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^Γ_α. □

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 Γ_A denote the smallest $\mathrm{FL_S}$-closed set containing A. We showed in Lemma 3.5.5 that the model $\mathcal{M}^\Gamma$ is a Γ_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 $\mathcal{M}^C, s \models \neg A$, so $\mathcal{M}^C, s \not\models A$. As $A \in \Gamma_A$, then by Lemma 3.5.5 $\mathcal{M}^\Gamma, [s] \not\models A$. By Lemma 3.5.6 $\mathcal{M}^\Gamma$ 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. □

3.6 Soundness and Completeness of SPDL+∪ and SPDL+*

It follows from the proofs of Theorems 3.2.1 and 3.4.1 that SPDL+∪ 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+$*$, 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 $V : \Phi \to \wp(S)$ 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 α_1 be a program of a logic $\mathcal{L}_1$ and α_2 be a program of a logic $\mathcal{L}_2$. We say α_1 and α_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 $R_{\alpha_1} = R_{\alpha_2}$.

Definition 4.1.3 Let $\mathcal{L}_1$, $\mathcal{L}_2$ be logics. We say that a formula F (or program α) of $\mathcal{L}_1$ *can be expressed* in $\mathcal{L}_2$ if there is a formula F' (or program α') of $\mathcal{L}_2$ which is equivalent to F (or α).

Definition 4.1.4 We say $\mathcal{L}_1$ and $\mathcal{L}_2$ have *equivalent expressive power* if every formula of $\mathcal{L}_1$ can be expressed in $\mathcal{L}_2$, and every formula of $\mathcal{L}_2$ can be expressed in $\mathcal{L}_1$. We say that $\mathcal{L}_1$ has *greater expressive power* than $\mathcal{L}_2$ if every formula of $\mathcal{L}_2$ can be expressed in $\mathcal{L}_1$ but there is a formula of $\mathcal{L}_1$ which cannot be expressed in $\mathcal{L}_2$. In this case $\mathcal{L}_2$ has *less expressive power* than $\mathcal{L}_1$.

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 σ^*, 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 α be a program of SPDL. We consider the possibilities for α:

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

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

$$\begin{aligned}
R_{\alpha'} &= \{(s,s) : \mathcal{M}, s \models_{\text{PDL}} \textbf{true}\} \\
&= \{(s,s) : s \in S\} \\
&= id_S \\
&= R_{\textbf{skip}} \\
&= R_\alpha
\end{aligned}$$

Case 3: $\alpha = \textbf{abort}$. Let $\alpha' = \textbf{false}? \in \mathcal{P}_{\mathrm{PDL}}$. Then

$$\begin{aligned} R_{\alpha'} &= \{(s,s) : \mathcal{M}, s \models_{\mathrm{PDL}} \textbf{false}\} \\ &= \emptyset \\ &= R_{\textbf{abort}} \\ &= R_\alpha \end{aligned}$$

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

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

$$\begin{aligned} R_{E'?} &= \{(s,s) : \mathcal{M}, s \models_{\mathrm{PDL}} E'\} \\ &= \{(s,s) : \mathcal{M}, s \models_{\mathrm{SPDL}} E\} \\ R_{E'?;\beta'} &= \{(s,s') : (s,s) \in R_{E'?} \text{ and } (s,s') \in R_{\beta'}\} \\ &= \{(s,s') : \mathcal{M}, s \models_{\mathrm{SPDL}} E \text{ and } (s,s') \in R_\beta\} \\ R_{(E'?;\beta')^*} &= \{(s,s') : (\exists n)(\exists s_0, \ldots s_n \in S)(s = s_0, s' = s_n, \text{ and} \\ &\quad (\forall j < n)((s_j, s_{j+1}) \in R_{E'?;\beta'}))\} \\ &= \{(s,s') : (\exists n)(\exists s_0, \ldots s_n \in S)(s = s_0, s' = s_n, \text{ and} \\ &\quad (\forall j < n)(\mathcal{M}, s_j \models_{\mathrm{SPDL}} E \text{ and } (s_j, s_{j+1}) \in R_\beta))\} \\ R_{\alpha'} &= R_{(E'?;\beta')^*;\neg E'?} \\ &= \{(s,s') : (\exists n)(\exists s_0, \ldots s_n \in S)(s = s_0, s' = s_n, \text{ and} \\ &\quad (\forall j < n)(\mathcal{M}, s_j \models_{\mathrm{SPDL}} E \text{ and } (s_j, s_{j+1}) \in R_\beta \text{ and } (s', s') \in R_{\neg E'?})\} \\ &= \{(s,s') : (\exists n)(\exists s_0, \ldots s_n \in S)(s = s_0, s' = s_n, \text{ and} \\ &\quad (\forall j < n)(\mathcal{M}, s_j \models_{\mathrm{SPDL}} E \text{ and } (s_j, s_{j+1}) \in R_\beta \text{ and } \mathcal{M}, s' \models_{\mathrm{SPDL}} \neg E)\} \\ &= R_{\textbf{while E do } \beta} \\ &= R_\alpha \end{aligned}$$

Case 6: α = **if** E **then** β **else** γ. By induction there exist a formula E' and programs β' and γ' of PDL which are equivalent to E, β and γ respectively. Let $\alpha' = (E'?;\beta') \cup (\neg E'?;\gamma')$. Then

$$\begin{aligned}
R_{\alpha'} &= R_{(E'?;\beta')\cup(\neg E'?;\gamma')} \\
&= R_{E'?;\beta'} \cup R_{\neg E'?;\gamma'} \\
&= \{(s,s') : \mathcal{M}, s \models_{\text{PDL}} E' \text{ and } (s,s') \in R_{\beta'}\} \\
&\quad \cup \{(s,s') : \mathcal{M}, s \models_{\text{PDL}} \neg E' \text{ and } (s,s') \in R_{\gamma'}\} \\
&= \{(s,s') : \mathcal{M}, s \models_{\text{SPDL}} E \text{ and } (s,s') \in R_{\beta}\} \\
&\quad \cup \{(s,s') : \mathcal{M}, s \models_{\text{SPDL}} \neg E \text{ and } (s,s') \in R_{\gamma}\} \\
&= R_{\textbf{if E then } \beta \textbf{ else } \gamma} \\
&= R_{\alpha}
\end{aligned}$$

□

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 $F = [(\sigma \cup \tau)^*](\langle\sigma\rangle\textbf{true} \wedge \langle\tau\rangle\textbf{true})$. This formula is satisfiable, and an example of a model for F is an infinite binary tree that can be drawn so that σ is the set of transitions from each node to its left hand child, and τ 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. □

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

Proof: This follows from Lemmas 4.2.1 and 4.2.2. $\square$

4.3 Extensions of SPDL

We now consider the extensions of SPDL by "$\cup$" and "$*$", the logics SPDL+$\cup$ and SPDL+$*$. We are interested in whether or not the expressive powers of SPDL+$\cup$ 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+$\cup$, which has greater expressive power than SPDL.

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

Lemma 4.3.1 *There are formulas G of SPDL+$\cup$ and H of SPDL+$*$ which cannot be expressed in SPDL.*

Proof: We use the result of Lemma 4.2.2 that the formula $[(\sigma \cup \tau)^*](\langle\sigma\rangle\mathbf{true} \wedge \langle\tau\rangle\mathbf{true})$ of PDL cannot be expressed in SPDL. We now consider the formulas
$G = [\mathbf{while}\ (\langle\sigma\rangle\mathbf{true} \wedge \langle\tau\rangle\mathbf{true})\ \mathbf{do}\ (\sigma \cup \tau)]\mathbf{false}$ and $H = [(\sigma^*; \tau^*)^*](\langle\sigma\rangle\mathbf{true} \wedge \langle\tau\rangle\mathbf{true})$. We show that G and H are each equivalent to $F = [(\sigma \cup \tau)^*](\langle\sigma\rangle\mathbf{true} \wedge \langle\tau\rangle\mathbf{true})$, and hence cannot be expressed in SPDL. F is satisfied if $(\langle\sigma\rangle\mathbf{true} \wedge \langle\tau\rangle\mathbf{true})$ is true after the program $(\sigma \cup \tau)$ has been repeated any finite number of times. G is satisfied if we can never reach a state in which $(\langle\sigma\rangle\mathbf{true} \wedge \langle\tau\rangle\mathbf{true})$ is false by repeating the program $(\sigma \cup \tau)$. Hence F and G are equivalent. To show that F and H are equivalent notice that the programs $(\sigma \cup \tau)^*$ and $(\sigma^*; \tau^*)^*$ are equivalent. $\square$

Thus both SPDL+$\cup$ and SPDL+$*$ have greater expressive power than SPDL. We are now interested in whether either of SPDL+$\cup$ and SPDL+$*$ have expressive power equivalent to PDL. It follows from Lemma 4.2.1 that every formula and program of SPDL+$\cup$ 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 α of PDL is in $\cup$-*normal form* if $\alpha = \beta_1 \cup \ldots \cup \beta_n$, where each β_j is constructed from atomic programs and tests using only the operators ";" and "$*$".

Lemma 4.3.3 *Every program α of PDL is equivalent to a program $\alpha_\cup$ in $\cup$-normal form.*

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

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

Case 2: $\alpha = F?$ for some $F \in \mathcal{F}_{\mathrm{PDL}}$.

Case 3: $\alpha = \alpha^1;\alpha^2$. By induction there exist $\alpha^i_\cup = \beta^i_1 \cup \ldots \cup \beta^i_{n_i}$ as above, for $i = 1, 2$. Then

$$\begin{aligned}
\alpha &= \alpha^1;\alpha^2 \\
&\equiv \alpha^1_\cup;\alpha^2_\cup \\
&= (\beta^1_1 \cup \ldots \cup \beta^1_{n_1});(\beta^2_1 \cup \ldots \cup \beta^2_{n_2}) \\
&\equiv (\beta^1_1;\beta^2_1) \cup (\beta^1_1;\beta^2_2) \cup \ldots \cup (\beta^1_{n_1};\beta^2_{n_2}).
\end{aligned}$$

Hence $\alpha_\cup = (\beta^1_1;\beta^2_1) \cup (\beta^1_1;\beta^2_2) \cup \ldots \cup (\beta^1_{n_1};\beta^2_{n_2})$.

Case 4: $\alpha = \alpha^1 \cup \alpha^2$.

Case 5: $\alpha = \delta^*$. By induction there exists $\delta_\cup = \beta_1 \cup \ldots \cup \beta_n$ as above. We use the result that $(\alpha \cup \beta)^* \equiv (\alpha^*;\beta^*)^*$.

$$\begin{aligned}
\alpha &= \delta^* \\
&\equiv \delta^*_\cup \\
&= (\beta_1 \cup \ldots \cup \beta_n)^* \\
&\equiv (\beta^*_1;(\beta_2 \cup \ldots \cup \beta_n)^*)^* \\
&\equiv (\beta^*_1;(\beta^*_2;(\beta_3 \cup \ldots \cup \beta_n)^*)^*)^* \\
&\equiv (\beta^*_1;(\beta^*_2;(\ldots\beta^*_n)^* \ldots)^*)^*.
\end{aligned}$$

Hence $\alpha_{\cup} = (\beta_1^*; (\beta_2^*; (\ldots \beta_n^*)^* \ldots)^*)^*$.

□

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 ∪-normal form. Hence we can assume $\alpha = \beta_1 \cup \ldots \cup \beta_n$ where each β_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 β_j there is a formula E' of SPDL + ∗ which is equivalent to E. Let β_j' be obtained from β_j by replacing each test $E?$ by (**if** E' **then skip else abort**). Then each β_j' is a program of SPDL + ∗, and $\alpha \equiv \beta_1' \cup \ldots \cup \beta_n'$. Hence

$$\begin{aligned} F &= [\alpha]G \\ &\equiv [\beta_1' \cup \ldots \cup \beta_n']G' \\ &\equiv [\beta_1']G' \wedge \ldots \wedge [\beta_n']G'. \end{aligned}$$

Thus $F' = [\beta_1']G' \wedge \ldots \wedge [\beta_n']G'$ is a formula of SPDL+∗ which is equivalent to F. □

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 $\mathcal{M}^{\cup} = \langle S, \{R_\alpha : \alpha \in \Pi\}, V\rangle$, 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).

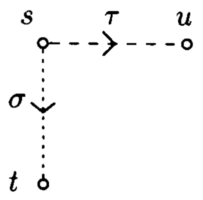

Figure 4.1: The model $\mathcal{M}^{\cup}$

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

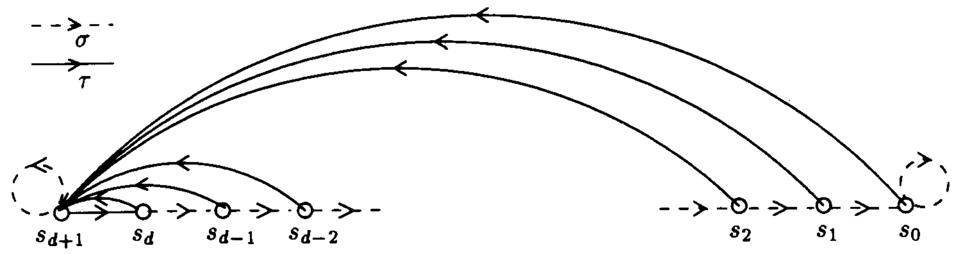

Figure 4.2: The segment model $\mathcal{S}_d$

For each $d \in N$ we define a *segment model* $\mathcal{S}_d$, 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 $\mathcal{S}_d$, identifying endpoints of adjacent copies, and adjoining an additional state $\tilde{s}$, as shown in Figure 4.3. Let $\tilde{s} \in V(p)$ for every $p \in \Phi$.

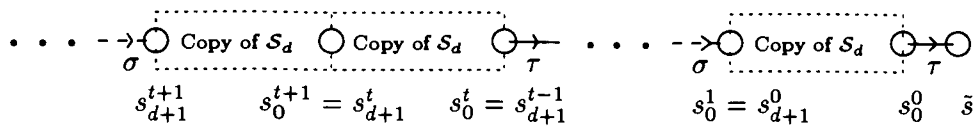

Figure 4.3: The model $\mathcal{M}_d$

Hence $\mathcal{M}_d = \langle S, \{R_\pi : \pi \in \Pi\}, V\rangle$ where $S = \{\tilde{s}, s^0_0, s^0_1, \ldots, s^0_{d+1} = s^1_0, s^1_1, \ldots\}$, R_σ and R_τ 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 $\overline{V}(G)$ and $\overline{V}(\neg G)$ are infinite in $\mathcal{M}_d$ for every $d \in N$.*

Proof: Consider the PDL-formula $\langle(\sigma^*;\tau;\sigma^*;\tau)^*\rangle[\sigma \cup \tau]$**false**. We show this formula is true in states s^t_0 when t is odd and false in states s^t_0 when t is even. We observe that $\mathcal{M}_d, s \models [\sigma \cup \tau]$**false** if and only if $s = \tilde{s}$, hence $\mathcal{M}_d, s^t_0 \models \langle(\sigma^*;\tau;\sigma^*;\tau)^*\rangle[\sigma \cup \tau]$**false** if and only if $s^t_0 R_{(\sigma^*;\tau;\sigma^*;\tau)^*}\tilde{s}$. From any state s^t_h with $h \leq d$, the program $\sigma^*;\tau$ is a transition to either s^{t-1}_d or $s^t_{d+1} = s^{t+1}_0$. From s^{t-1}_d the program $\sigma^*;\tau$ is a transition to either s^{t-2}_d or s^t_0, and from s^{t+1}_0 the program $\sigma^*;\tau$ is a transition to either s^t_d or s^{t+2}_0. Hence from state s^t_h with $h \leq d$ the program $\sigma^*;\tau;\sigma^*;\tau$ is a transition to one of s^{t-2}_d, s^t_0, s^t_d, or s^{t+2}_0. Notice that $sR_{(\sigma^*;\tau;\sigma^*;\tau)^*}\tilde{s}$ if and only if $s = s^1_h$ for some $h \leq d$. Hence $s^t_0 R_{(\sigma^*;\tau;\sigma^*;\tau)^*}\tilde{s}$ if and only if $t \equiv 1 \bmod 2$ i.e. if and only if t is odd. Hence the formula is true in infinitely many states and false in infinitely many states. $\square$

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

Definition 4.3.7 An SPDL+$\cup$ program α 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) γ_{jk} is a test, i.e. of the form **if** E **then skip else abort**, abbreviated $E?$

iii) $\gamma_{jk} =$ **while** E_{jk} **do** δ_{jk}.

Lemma 4.3.8 *Every program α of SPDL+$\cup$ is equivalent to a program α_N which is in normal form.*

Proof: Use induction on programs.

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

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

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

Case 4: $\alpha = \alpha^1;\alpha^2$. Suppose α^i has normal form $\alpha^i_N = \beta^i_1 \cup \ldots \cup \beta^i_{n_i}$. Then

$$\begin{aligned}
\alpha &= \alpha^1;\alpha^2 \\
&\equiv \alpha^1_N;\alpha^2_N \\
&= (\beta^1_1 \cup \ldots \cup \beta^1_{n_1});(\beta^2_1 \cup \ldots \cup \beta^2_{n_2}) \\
&\equiv (\beta^1_1;\beta^2_1) \cup (\beta^1_1;\beta^2_2) \cup \ldots \cup (\beta^1_1;\beta^2_{n_2}) \cup \ldots \cup (\beta^1_{n_1};\beta^2_{n_2}).
\end{aligned}$$

Hence $\alpha_N = (\beta^1_1;\beta^2_1) \cup (\beta^1_1;\beta^2_2) \cup \ldots \cup (\beta^1_1;\beta^2_{n_2}) \cup \ldots \cup (\beta^1_{n_1};\beta^2_{n_2})$.

Case 5: $\alpha =$ **while** E **do** δ.

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

$$\begin{aligned}
\alpha &= \mathbf{if}\ E\ \mathbf{then}\ \delta^1\ \mathbf{else}\ \delta^2 \\
&\equiv (E?;\delta^1) \cup (\neg E?;\delta^2) \\
&\equiv (E?;\delta^1_N) \cup (\neg E?;\delta^2_N) \\
&\equiv (E?;(\beta^1_1 \cup \ldots \cup \beta^1_{n_1})) \cup (\neg E?;(\beta^2_1 \cup \ldots \cup \beta^2_{n_2})) \\
&\equiv ((E?;\beta^1_1) \cup (E?;\beta^1_2) \cup \ldots \cup (E?;\beta^1_{n_1})) \cup ((\neg E?;\beta^2_1) \cup \ldots \cup (\neg E?;\beta^2_{n_2})) \\
&\equiv (E?;\beta^1_1) \cup (E?;\beta^1_2) \cup \ldots \cup (E?;\beta^1_{n_1}) \cup (\neg E?;\beta^2_1) \cup \ldots \cup (\neg E?;\beta^2_{n_2}).
\end{aligned}$$

Hence $\alpha_N = (E?;\beta^1_1) \cup (E?;\beta^1_2) \cup \ldots \cup (E?;\beta^1_{n_1}) \cup (\neg E?;\beta^2_1) \cup \ldots \cup (\neg E?;\beta^2_{n_2})$.

Case 7: $\alpha = \alpha^1 \cup \alpha^2$.

□

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

We say a formula F is *finite* (*cofinite*) in $\mathcal{M}_d$ if $\overline{V}(F)$ is finite (cofinite) in $\mathcal{M}_d$.

In the next lemma we prove that for any formula F of SPDL + $\cup$ we can find d such that F is either finite or cofinite in $\mathcal{M}_d$. We in fact show that F is either finite or cofinite in $\mathcal{M}_d$ for all sufficiently large d. To prove this we also show that for every program of the form "**while** E **do** α" the sets $R_{E,\alpha}(s)$ 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 $\mathcal{M}_d$ whenever d is sufficiently large, Wilm shows there exists $t \in N$ such that F is either true in every state above s_0^t in $\mathcal{M}_d$ or false in every state above s_0^t in $\mathcal{M}_d$ whenever d is sufficiently large. For the assertion about **while**-programs we will show that for all but finitely many states the sets $R_{E,\alpha}(s)$ 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_0^t in $\mathcal{M}_d$ whenever d is sufficiently large. We now state and prove the lemma.

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

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

(P)(i): *$\{s : R_{E,\alpha}(s) \neq \{s\}\}$ is finite.*

(P)(ii): *there is a finite set $S_{E,\alpha}$ of states of $\mathcal{M}_d$ such that for all but finitely many $s \in S$, $R_{E,\alpha}(s) = S_{E,\alpha}$.*

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 $\mathcal{M}_d$ we have $\overline{V}(F) = S$, hence F is cofinite in $\mathcal{M}_d$ 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 $\mathcal{M}_d$ for all $d \geq d_G$. Set $d_F = d_G$, and let $d \geq d_F$. F is finite in $\mathcal{M}_d$ if G is cofinite, and cofinite if G is finite.

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

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

Case 4.1: $\alpha' \in \Pi$, so α' 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 $\mathcal{M}_d$ for $d \geq 0$. For $\alpha' = \sigma, \tau$ note that for any state s both σ and τ are transitions to a state at most $d + 1$ states away. Hence if E is finite (cofinite) in $\mathcal{M}_d$ then so are $[\sigma]E$ and $[\tau]E$. Hence $[\alpha']E$ is finite or cofinite in $\mathcal{M}_d$ for $d \geq d_{[\alpha']E} = d_E$.

Case 4.2: $\alpha' = \textbf{skip}$. Suppose E is finite or cofinite in $\mathcal{M}_d$ for every $d \geq d_E$. We know $\models [\textbf{skip}]E \leftrightarrow E$ 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 $\mathcal{M}_d$ if E is.

Case 4.3 $\alpha' = \textbf{abort}$. Since $R_{\textbf{abort}} = \emptyset$ we have $\models [\textbf{abort}]E$ for any formula E, hence $[\alpha']E$ is cofinite in $\mathcal{M}_d$ 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 $\mathcal{M}_d$ 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 $\mathcal{M}_d$ 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 $\mathcal{M}_d$ for $d \geq d_{[\beta]E'}$, and $[\gamma]E'$ is either finite or cofinite in $\mathcal{M}_d$ 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** β. 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 $\{s : R_{H,\beta}(s) \neq R_{\text{skip}}(s)\}$ is finite, and there is $t_{H,\beta} \in N$ such that $R_{H,\beta}(s^t_h) = R_{\text{skip}}(s^t_h) = \{s^t_h\}$ for every $t \geq t_{H,\beta}$ and $h \leq d+1$. Note that $\{s^h_t : t < t_{H,\beta}\}$ 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
$\{s^t_h : t \geq t_{H,\beta}, h \leq d+1, \mathcal{M}_d, s^t_h \models [$**while** H**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 $s \in S$, $R_{H,\beta}(s) = S_{H,\beta}$. Hence there is $t_{H,\beta}$ such that $R_{H,\beta}(s^h_t) = S_{H,\beta}$ for every $t \geq t_{H,\beta}$ and $h \leq d+1$. Again note that $\{s^h_t : t < t_{H,\beta}\}$ is finite. If $\mathcal{M}, s \models E$ for every $s \in S(H,\beta)$ then $\mathcal{M}, s^t_h \models [$**while** E **do** $\beta]E$ for every state s^h_t 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 $\mathcal{M}, s^t_h \models \neg[$**while** E **do** $\beta]E$ for every state s^h_t 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** β **else** γ. From the axiom schema COND we know $\models [$**if** H **then** β **else** $\gamma]E \leftrightarrow ((H \rightarrow [\beta]E) \wedge (\neg H \rightarrow [\gamma]E))$ 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 $d \geq d_{[\alpha']E}$, $[$**if** H **then** β **else** $\gamma]E$ is either finite or cofinite in $\mathcal{M}_d$.

Case 4.7: $\alpha' = \beta \cup \gamma$. We know $[\beta \cup \gamma]E \leftrightarrow ([\beta]E \wedge [\gamma]E)$ 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 $([\beta]E \wedge [\gamma]E) = [\alpha']E$.

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

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

$$\begin{aligned} T_j &= \{F : \text{ either } F? = \gamma_{ji} \text{ or } \neg F? = \gamma_{ji} \text{ for some } i, 1 \leq i \leq m_j\} \\ d_{T_j} &= \max\{d_F : F \in T_j\}. \end{aligned}$$

We can now define

$$\begin{aligned} T &= \cup_{j \leq n} T_j \\ d_T &= \max\{d_{T_j} : 1 \leq j \leq n\}. \end{aligned}$$

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

If γ_{ji} is the program **while** F **do** δ 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 $\mathcal{M}_d$. If some γ_{ji} is a **while**-program then we can define the following

$$\begin{aligned} W_j &= \{(\textbf{while } F \textbf{ do } \delta) : \gamma_{ji} = (\textbf{while } F \textbf{ do } \delta) \text{ for some } i,\ 1 \leq i \leq m_j\} \\ d_{W_j} &= \max\{d_{F,\delta} : (\textbf{while } F \textbf{ do } \delta) \in W_j\}. \end{aligned}$$

We can now define

$$\begin{aligned} W &= \cup_{j \leq n} W_j \\ d_W &= \max\{d_{W_j} : 1 \leq j \leq n\}. \end{aligned}$$

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

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

$$d_{E,\alpha} \quad = \quad \max\{d_E, d_T, d_W, 2f_\sigma + 1\}$$

Let $d \geq d_{E,\alpha}$ 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 **while**-program in W. Let $\tilde{t}$ be such that E, and every $F \in T$, are either true in every state s^t_h with $t \geq \tilde{t}$, or false in every state s^t_h with $t \geq \tilde{t}$, and for every **while**-program in W either $R_{F,\delta}(s^t_h) = \{s^t_h\}$ for every state s^t_h with $t \geq \tilde{t}$, or $R_{F,\delta}(s^t_h) = S_{F,\delta}$ for every state s^t_h with $t \geq \tilde{t}$. Set $t_{E,\alpha} = \tilde{t} + 1$. Then for every $d \geq d_{E,\alpha}$, each test occuring in one of the β_j's is either equivalent to **skip** in every state s^t_h with $t \geq t_{E,\alpha} - 1$, and $h \leq d + 1$, or equivalent to **abort** in every state s^t_h with $t \geq t_{E,\alpha} - 1$, and $h \leq d + 1$. Every **while**-program is either equivalent to **skip** in every state s^t_h with $t \geq t_{E,\alpha} - 1$, and $h \leq d + 1$, or a transition to a state below $s_0^{t_{E,\alpha}}$ in every state s^t_h with $t \geq t_{E,\alpha} - 1$, and $h \leq d + 1$. Hence for each β_j which has non-aborting executions in state $s_0^{t_{E,\alpha}}$ or some state above $s_0^{t_{E,\alpha}}$ there is a sequence of occurences of σ, τ, **skip** and **while**-programs for which (P)(ii) holds, such that any execution of β_j in or above state $s_0^{t_{E,\alpha}}$ is equivalent to an execution of the concatenation of this sequence until a state below $s_0^{t_{E,\alpha}}$ is reached.

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

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

Case 2: E is cofinite in $\mathcal{M}_d$. For every $t \geq t_{E,\alpha} - 1$ and $h \leq d + 1$ we have $\mathcal{M}, s^t_h \models E$, and each β_j is equivalent at s^t_h to a sequence of occurences of σ, τ and programs of the form "**while** F **do** δ" as described above. Suppose for some fixed $t' \geq t_{E,\alpha}$ and $h' \leq d + 1$ that $s^{t'}_{h'} R_{\textbf{while } E \textbf{ do } \alpha} s'$ where s' is a state in $\mathcal{M}_d$. We show that $s^t_h R_{\textbf{while } E \textbf{ do } \alpha} s'$ for every $t \geq t_{E,\alpha}$ and $h \leq d + 1$.

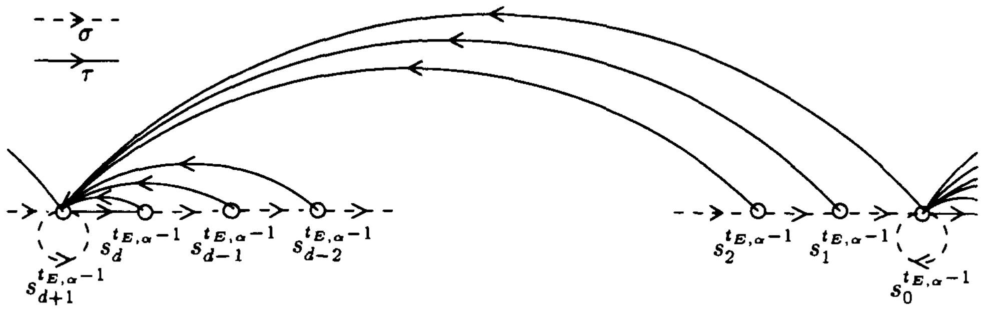

Figure 4.4: The buffer zone

We consider the section of $\mathcal{M}_d$ between states $s_{d+1}^{t_{E,\alpha}-1}$ and $s_0^{t_{E,\alpha}-1}$, shown in Figure 4.4. We call this the *buffer zone*. As $s_{h'}^{t'} R_{\textbf{while } E \textbf{ do } \alpha} s'$ there is a sequence $\beta_{j1}, \ldots, \beta_{jk}$ such that $s_{h'}^{t'} R_{\beta_{j1};\ldots;\beta_{jk}} s'$, and we know $\mathcal{M}, s' \models \neg E$, so s' must be below state $s_0^{t_{E,\alpha}-1}$. For each program "**while** F **do** δ" which occurs in one of the β_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** δ" for which (P)(ii) holds occurs in any of $\beta_{j1}, \ldots, \beta_{jk}$, or the system is at a state below $s_0^{t_{E,\alpha}-1}$ 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 β_j is equivalent (until a state below $s_0^{t_{E,\alpha}-1}$ is reached) to a sequence of occurrences of σ and τ as described above, and $\mathcal{M}_d, s_h^{t_{E,\alpha}-1} \models E$ for every $h \leq d+1$. Also $s_{h'}^{t'}$ 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 $s_{h'}^{t'}$ 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 β_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 σ. As we chose $d_{E,\alpha} \geq 2f_\sigma + 1$ none of the programs β_j, $1 \leq j \leq n$, satisfy this condition.

Observation 2: there is a program β_k which is equivalent to a sequence of occurrences of σ 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 occurence of τ 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 β_j contains an occurence of τ 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 $\beta_{j1}; \ldots ; \beta_{jk}$ 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 β_l with an odd number of occurrences of τ.

Justification: Consider a program β_l' in the sequence $\beta_{j1}; \ldots ; \beta_{jk}$, such that the system is in or above state $s_{d+1}^{t_{E,\alpha}-1}$ before executing β_l' and in a state below $s_{d+1}^{t_{E,\alpha}-1}$ after executing β_l'. The first occurence of τ in this program must occur at state $s_{d+1}^{t_{E,\alpha}-1}$, as if it occured above $s_{d+1}^{t_{E,\alpha}-1}$ then there would be a transition to $s_{d+1}^{t_{E,\alpha}}$ and, by Observation 1, β_l' couldn't reach a state below $s_0^{t_{E,\alpha}} = s_{d+1}^{t_{E,\alpha}-1}$. So the first occurence of τ is a transition from $s_{d+1}^{t_{E,\alpha}-1}$ to $s_d^{t_{E,\alpha}-1}$. Now suppose there is a second occurence of τ. 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 τ, which must happen as β_l' ends in a state below $s_{d+1}^{t_{E,\alpha}-1}$. Clearly β_l' must contain an odd number of occurences of τ. Set $\beta_l = \beta_l'$.

Observation 4: There must be a state $s_{h''}^{t_{E,\alpha}-1}$, $0 < h'' \leq d$ such that $s_{h''}^{t_{E,\alpha}-1} R_{\mathbf{while}\ E\ \mathbf{do}\ \alpha} s'$.

Justification: The final state reached by the program β_l 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 $s_{h''}^{t_{E,\alpha}-1} R_{\mathbf{while}\ E\ \mathbf{do}\ \alpha} s'$.

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 β_j $(1 \leq j \leq n)$ such that $s_h^t R_{\alpha_{t,h}} s_{h''}^{t_{E,\alpha}-1}$.

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 β_k enough (at most d) times to reach state s_0^t, then use β_l to reach a state s_h^{t-1}. Repeat this procedure to reach s_h^{t-2}, then s_h^{t-3} and so on until state $s_h^{t_{E,\alpha}}$ is reached. Repeating β_k another d times we reach $s_0^{t_{E,\alpha}}$, then β_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 $s_h^t R_{\alpha_{t,h}} s_h^{t_{E,\alpha}-1} R_{\mathbf{while}\ E\ \mathbf{do}\ \alpha} s'$ and as $\alpha_{t,h}$ is a the concatenation of a finite sequence of programs β_j $(1 \leq j \leq n)$ it follows that $s^t_h R_{\textbf{while } E \textbf{ do } \alpha} s'$, 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 $\mathcal{M}_d$. As each state s in $S(E,\alpha)$ is below $s_0^{t_{E,\alpha}-1}$, and there are only finitely many states below $s_0^{t_{E,\alpha}-1}$, then $S(E,\alpha)$ must be finite and (P)(ii) holds for $\mathcal{M}_d$.

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 $s_0^{t_{E,\alpha}-1}$. Let "**while** F **do** δ" be the first such program to occur, and suppose it occurs in β_i, when the system is in state $s^{t''}_{h''}$ for some $t'' \geq t_{E,\alpha}-1$ and $h'' \leq d+1$. Let k be such that γ_{ik} is the first occurrence of "**while** F **do** δ" in β_i. Set $\gamma' = \gamma_{i1};\ldots;\gamma_{i,k-1}$ and $\gamma'' = \gamma_{i,k+1};\ldots;\gamma_{im_i}$, 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 $s^{t'}_{h'} R_{\textbf{while } E \textbf{ do } \alpha} s'$ we must have $s R_{\textbf{while } E \textbf{ do } \alpha} s'$ for some $s \in S(F,\delta)$, and there is a state s'' such that $s R_{\gamma''} s'' R_{\textbf{while } E \textbf{ do } \alpha} s'$.

We now consider a general state s^t_h in $\mathcal{M}_d$, where $t \geq t_{E,\alpha}$ and $h \leq d+1$. At this state γ' is equivalent to a sequence of occurrences of σ and τ, and can only reach states above $s_0^{t_{E,\alpha}-1}$ (as s^t_h is above $s_d^{t_{E,\alpha}-1}$ and, by Observation 1 proved for the base case, none of the β_j's can be a transition from a state above $s_d^{t_{E,\alpha}-1}$ to a state below $s_0^{t_{E,\alpha}-1}$). So there is a state $s^{t''}_{h''}$ such that $s^t_h R_{\gamma'} s^{t''}_{h''}$ and $s^{t''}_{h''}$ is above $s_0^{t_{E,\alpha}-1}$, which means $S^{t''}_{h''}(F,\delta) = S(F,\delta)$. In particular $s^{t''}_{h''} R_{\textbf{while } F \textbf{ do } \delta} s$, where s is as above, so $s R_{\gamma''} s'' R_{\textbf{while } E \textbf{ do } \alpha} s'$. Hence $s^t_h R_{\gamma'} s^{t''}_{h''} R_{\textbf{while } F \textbf{ do } \delta} s R_{\gamma''} s'' R_{\textbf{while } E \textbf{ do } \alpha} s'$, so $s^t_h R_{\beta_i} s'' R_{\textbf{while } E \textbf{ do } \alpha} s'$, which yields $s^t_h R_{\textbf{while } E \textbf{ do } \alpha} s'$ which is what was required. Hence there is a set of states $S(E,\alpha)$ such that $S^t_h(E,\alpha) = S(E,\alpha)$ for every $t \geq t_{E,\alpha}$ and $h \leq d+1$. Again we observe that there are only finitely many states below $s_0^{t_{E,\alpha}-1}$ so $S(E,\alpha)$ must be finite, and (P)(ii) holds.

□

Lemma 4.3.10 *There is a formula of PDL which cannot be expressed in SPDL+*U.

Proof: It follows from Lemma 4.3.9 that for every formula F of SPDL+U there is a d such that either $\overline{V}(F)$ or $\overline{V}(\neg F)$ is finite in $\mathcal{M}_d$. 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 $\mathcal{M}_d$ for every $d \in N$, hence G is a formula of PDL which cannot be expressed in SPDL+$\cup$. □

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

[1] FISCHER, M.J. AND LADNER, R.E. *Propositional modal logic of programs*, **Proceedings of the 9th Annual ACM Symposium on Theory of Computing** (1977), pp. 286–294.

[2] GOLDBLATT, ROBERT, **Axiomatising the Logic of Computer Programming**, Springer-Verlag, Berlin Heidelberg, 1982.

[3] GOLDBLATT, ROBERT, *The semantics of Hoare's iteration rule*, **Studia Logica 41** (1982), pp. 141–158.

[4] GOLDBLATT, ROBERT, **Logics of Time and Computation**, Centre for the Study of Language and Information, Stanford, 1992.

[5] HALPERN, J.Y. AND REIF, J.H., *The propositional dynamic logic of deterministic, well-structured programs*, **Proceedings of the 22nd Annual IEEE Symposium on Foundations of Computer Science** (1981), pp. 322–334.

[6] HAREL, D. *Dynamic logic*, **Handbook of Philosophical Logic** Vol. 2, Gabbay, D. and Guenthner, F. eds, Reidel, Dordrecht, 1984, pp. 497–604.

[7] HUGHES, G.E. AND CRESSWELL, M.J., **An Introduction to Modal Logic**, Routledge, London, 1968.

[8] HUGHES, G.E. AND CRESSWELL, M.J., **A Companion to Modal Logic**, Methuen, London, 1984.

[9] PRATT, V.R., *Semantical considerations on Floyd-Hoare logic*, **Proceedings of 17th Annual IEEE Symposium in Foundations of Computer Science** (1976), pp. 109–121.

[10] WILM, ANDREAS, *Determinism and non-determinism in PDL*, **Theoretical Computer Science 87** (1991), pp. 189–202.