A Canonical Model Construction for Iteration-Free PDL with Intersection

Florian Bruse Daniel Kernberger
Martin Lange
School of Electr. Eng. and Computer Science, University of Kassel, Germany

We study the axiomatisability of the iteration-free fragment of Propositional Dynamic Logic with Intersection and Tests. The combination of program composition, intersection and tests makes its proof-theory rather difficult. We develop a normal form for formulae which minimises the interaction between these operators, as well as a refined canonical model construction. From these we derive an axiom system and a proof of its strong completeness.

1 Introduction

Propositional Dynamic Logic (PDL) is a multi-modal logic with a two-sorted language. It defines *formulae* and *programs*. Formulae make assertions about worlds in a Kripke structure referring to the ability to access other worlds using programs. The term "program" originates from PDL's early use in program specification and verification. It can be seen as a propositional Floyd-Hoare calculus [12]. PDL is also related to logics used in knowledge representation, it is basically the same as the description logic $\mathcal{ALC}_{\mathsf{reg}}$ [14].

PDL is a well-behaved modal logic in that its satisfiability problem is decidable. The first upper bound result yielded NEXPTIME based on a small model property and polynomial time model checking [7]. Later the problem was shown to be EXPTIME-complete [13, 7]. Axiomatisations for PDL have also been given, for instance in form of the Segerberg axioms [10] and others, c.f. [8].

In plain PDL, the programs form a Kleene algebra, built from atomic programs with the operations *union*, *composition* and *iteration*. A natural question concerns its extension with other program constructs and its effect on expressive power, decidability, complexity, axiomatisability, etc. One such operator is *test* which creates new basic programs from formulae (hence making them mutually recursive). PDL with Tests ($\mathrm{PDL}^{?}$) is more expressive than PDL [5] but satisfiability has the same complexity. Thus, adding just tests does not create conceptual problems; it only tends to complicate correctness proofs slightly. Other operators that have been found to be equally harmless in this respect are *looping* [15] and *converse* [7].

Another program construct that one may consider naturally is *intersection*. PDL with Intersection ($\mathrm{PDL}^{\cap}$) turns out to be more complex than PDL. Its satisfiability problem is – perhaps surprisingly – decidable, but it is 2EXPTIME-complete [6, 9]. The addition of intersection causes the loss of bisimulation-invariance and, hence, the tree model property, but it preserves a DAG model property. This does not hold true in the presence of tests anymore, though. $\mathrm{PDL}^{\cap,?}$ can require models to have (nested!) cycles.

The intersection operator also turns out to be intriguing for the problem of axiomatisability. There are studies concerning sound and complete axiomatisiations for versions of PDL with Intersection. Balbiani and Fariñas del Cerro axiomatised the small fragment without tests, Kleene stars and union [2]. Passy and Tinchev have shown that $\mathrm{PDL}^{\cap,?}$ without the Kleene star ($\mathrm{PDL}_0^{\cap,?}$) is axiomatisable in the stronger language of added nominals [11]. Nominals can be seen as atomic propositions that hold true in a unique

D. Cantone and G. Delzanno (Eds.): Seventh Symposium on
Games, Automata, Logics and Formal Verification (GandALF'16)
EPTCS 226, 2016, pp. 120–134, doi:10.4204/EPTCS.226.9

world of a Kripke structure. At close inspection one can see that the concept of naming particular worlds – which is known to break bisimulation-invariance as well – is very helpful on the way to a sound and complete axiomatisation for PDL with Intersection: as we will see in Sect. 3, the intersection operator can be used to require several copies of worlds that should otherwise look exactly the same. Nominals for instance can help to distinguish these copies.

Surely, an axiomatisation of a logic that uses features which are not available in the logic can be questioned. Balbiani and Vakarelov have re-considered the problem of axiomatisation for $\mathrm{PDL}_0^{\cap,?}$ and proposed a deductive system which only use logical *operators* that are available in the logic [3, 1]. However, the axiomatisation requires a larger vocabulary than the logic to be axiomatised has, see Sect. 3 for details. In [4], Balbiani and Vakarelov extended their work to full $\mathrm{PDL}^{\cap,?}$, but the issues with the differences between the object logic and the proof logic persist. The work that is closest to the one presented here is Balbiani's refinement of the $\mathrm{PDL}_0^{\cap,?}$ axiomatisation [1]. That calculus does not seem to rely on additional features outside of the object logic. Working out the exact connection between that axiomatisation and the one presented here is left as future work, see also the concluding remarks at the end of this paper.

Here we propose a normal form for $\mathrm{PDL}_0^{\cap,?}$ formulae which minimises the interaction of intersections and tests in its programs. We then present a canonical model construction for $\mathrm{PDL}_0^{\cap,?}$ which starts with all maximally consistent sets and introduces abstract accessibility relations between them that correspond to non-atomic programs. These abstract edges then get refined, possibly introducing new copies of maximally consistent sets until the model is saturated. Then every maximally consistent set is satisfied somewhere in this limit model, so we can derive a complete axiomatisation for $\mathrm{PDL}_0^{\cap,?}$ over any set of propositions $\mathcal{P}$ which does not need additional operators or propositions.

2 Iteration-Free PDL with Intersection and Tests

We fix a set $\mathcal{P} = \{p, q, \ldots\}$ of unary relation symbols called *atomic propositions* and a set $\mathcal{R} = \{a, b, \ldots\}$ of binary relation symbols called *atomic programs* for the rest of the paper. We refer to $\mathcal{P}, \mathcal{R}$ as the vocabulary τ.

Formulae φ and programs α of Iteration-Free Propositional Dynamic Logic with Intersection and Tests ($\mathrm{PDL}_0^{\cap,?}$) are derived by

$$\varphi \;\coloneqq\; p \mid \varphi \vee \varphi \mid \neg\varphi \mid \langle\alpha\rangle\varphi \qquad\qquad \alpha \;\coloneqq\; a \mid \alpha;\alpha \mid \alpha \cup \alpha \mid \alpha \cap \alpha \mid \varphi?$$

where $p \in \mathcal{P}, a \in \mathcal{R}$. Other Boolean and modal connectives like $\top, \bot, \wedge, \rightarrow, \leftrightarrow, [\alpha]$ can be derived as usual. For ease of notation and for reasons of readability, we sometimes use the abbreviation $\alpha^{\circlearrowleft} \coloneqq \alpha \cap \top?$. We use $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$ as the descending order of precedence in formulae and ;, $\cap$, $\cup$ in programs in order to save parentheses. Unary operators always bind stronger than binary ones.

Formulae are interpreted in worlds of a Kripke structure $\mathfrak{A}$; programs are interpreted as binary relations in these structures. A Kripke structure $\mathfrak{A}$ over τ is a set of worlds W together with interpretations $P_p^{\mathfrak{A}} \subseteq A$ for all $p \in \mathcal{P}$ and $R_a^{\mathfrak{A}} \subseteq A \times A$ for all $a \in \mathcal{R}$. A pointed Kripke structure with distinguished world u is written as $\mathfrak{A}, u$. The semantics is given inductively as follows.

$$\begin{array}{lll}
\mathfrak{A},u \models p & \text{iff} & u \in P_p^{\mathfrak{A}}, p \in \mathcal{P} \\
\mathfrak{A},u \models \varphi \vee \varphi' & \text{iff} & \mathfrak{A},u \models \varphi \text{ or } \mathfrak{A},u \models \varphi' \\
\mathfrak{A},u \models \neg\varphi & \text{iff} & \mathfrak{A},u \not\models \varphi \\
\mathfrak{A},u \models \langle\alpha\rangle\varphi & \text{iff} & \exists v.(u,v) \in R_\alpha^{\mathfrak{A}} \text{ and } \mathfrak{A},v \models \varphi
\end{array}
\qquad
\begin{array}{rcl}
R_{\varphi?}^{\mathfrak{A}} & \coloneqq & \{(u,u) \mid \mathfrak{A},u \models \varphi\} \\
R_{\alpha\cup\beta}^{\mathfrak{A}} & \coloneqq & R_\alpha^{\mathfrak{A}} \cup R_\beta^{\mathfrak{A}} \\
R_{\alpha\cap\beta}^{\mathfrak{A}} & \coloneqq & R_\alpha^{\mathfrak{A}} \cap R_\beta^{\mathfrak{A}} \\
R_{\alpha;\beta}^{\mathfrak{A}} & \coloneqq & R_\alpha^{\mathfrak{A}} \circ R_\beta^{\mathfrak{A}}.
\end{array}$$

We say that φ and ψ are equivalent and write $\varphi \equiv \psi$ if, for all pointed Kripke structures $\mathfrak{A}, u$, we have $\mathfrak{A}, u \vDash \varphi$ if and only if $\mathfrak{A}, u \vDash \psi$.

We write $\mathfrak{A}, u \vDash \Phi$ if $\mathfrak{A}, u \vDash \varphi$ for all $\varphi \in \Phi$. We write $u \xrightarrow{\alpha} v$ to indicate that $(u, v) \in R^{\mathfrak{A}}_\alpha$. We write further $\alpha \Rightarrow \beta$ if for all Kripke structures $\mathfrak{A}$ and all worlds u, v it holds that if $u \xrightarrow{\alpha} v$, then it also holds that $u \xrightarrow{\beta} v$.

Definition 1. Let α be a program and let $\mathfrak{A}$ be a Kripke structure in which $u \xrightarrow{\alpha} v$ for some u, v. A *witness graph* for $u \xrightarrow{\alpha} v$ is defined inductively over α:

- If $\alpha = a$ for $a \in \mathcal{R}$ then a witness graph consists of nodes u and v together with the a-edge between u and v.
- If $\alpha = \varphi?$ then a witness graph is the node $u (= v)$.
- If $\alpha = \alpha_1; \alpha_2$ then there is w such that $u \xrightarrow{\alpha_1} w$ and $w \xrightarrow{\alpha_2} v$. The witness graph for $u \xrightarrow{\alpha} v$ is the union of the witness graphs for $u \xrightarrow{\alpha_1} w$ and $w \xrightarrow{\alpha_2} v$.
- If $\alpha = \alpha_1 \cap \alpha_2$ then there are witness graphs for $u \xrightarrow{\alpha_1} v$ and $u \xrightarrow{\alpha_2} v$, and their union is the witness graph for $u \xrightarrow{\alpha} v$.
- If $\alpha = \alpha_1 \cup \alpha_2$ then there is a witness graphs for $u \xrightarrow{\alpha_1} v$ or for $u \xrightarrow{\alpha_2} v$. Either of these is a witness graph for $u \xrightarrow{\alpha} v$.

Witness graphs need not be unique. If $u \xrightarrow{\alpha} v$, there can be many witness graphs in a particular Kripke structure.

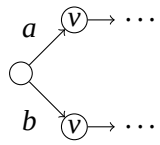
3 Canonical Models

A *canonical model* (for a modal logic) is typically a Kripke structure whose worlds are all the maximally consistent sets of the underlying logic whith respect to some notion of provability $\vdash$. Such a set Φ of formulae is *consistent* if it is not possible to derive a contradiction from it, i.e. if $\Phi \nvdash \bot$. It is a *maximally consistent set* (MCS) if it is consistent and maximal with respect to $\subseteq$, i.e. it is not possible to add any formula of the underlying logic without making it inconsistent. Canonical models are typically used to prove (strong) completeness of the axiomatisation $\vdash$ along the following lines. Suppose $\Phi \nvdash \bot$, i.e. Φ is consistent, then it is included in an MCS Φ'. Next one shows that for every world v in the canonical model $\mathfrak{K}$ and every formula φ of the underlying logic we have $\mathfrak{K}, v \vDash \varphi$ iff $\varphi \in v$. So a consistent set, and particularly an MCS, is satisfiable in the canonical model. Equally, every valid set of formulae is provable, i.e. the axiomatisation is complete.

Canonical models and PDL$_0^{\cap,?}$. A simple consequence of the standard understanding of a canonical model is the fact that no two worlds in it represent the same MCS. It is important to understand that no such standard canonical model construction can be used to prove completeness of an axiomatisation for PDL$_0^{\cap,?}$ as the following example shows.

Example 1. Let Φ be a satisfiable set of PDL$_0^{\cap,?}$ formulae. Consider the set

$$\mathit{Split}(\Phi) \;\coloneqq\; \{\langle a\rangle\top, \langle b\rangle\top, [a \cap b]\bot\} \cup \{[a]\varphi, [b]\varphi \mid \varphi \in \Phi\}\,.$$



It is easily seen to be satisfiable, too. Suppose $\mathfrak{A}, v$ is a model of Φ. A model for $\mathit{Split}(\Phi)$ is obtained using two disjoint copies of $\mathfrak{A}, v$ as shown on the right. It is equally possible to see that the two copies cannot be merged since this would contradict the requirement $[a \cap b]\bot$.

Canonical model constructions for $\mathrm{PDL}_0^{\cap,?}$ in the literature. As mentioned in the introduction, the literature contains proposals for $\mathrm{PDL}_0^{\cap,?}$ axiomatisations, most notably by Balbiani and Vakarelov [3, 1]. The intricacies introduced by program intersection are tackled using the following principle, c.f. [3, Prop. 2.1]: if u has an $(\alpha \cap \beta)$-successor then it has an α-successor v and a β-successor w such that v and w cannot be distinguished by any atomic proposition p. It is important to note that p is not restricted to be drawn from any pre-given set; instead it ranges over all propositions that could possibly *extend* an underlying model.

Balbiani and Vakarelov then formulate this semantic principle syntactically as a proof rule (INT) and present a refined canonical model construction that circumvents the problem with intersection as outlined in Ex. 1 as follows. Worlds of the canonical model are not MCS but *maximally consistent theories* (MCT). An MCT is an MCS that is closed under applications of rule (INT). Hence, every MCT is an MCS, and every MCS with this additional closure property is an MCT. Intuitively, closure under rule (INT) helps with the construction of a canonical model for formulae like the ones in Ex. 1 by introducing a new atomic proposition which can be used to distinguished two copies of worlds that would otherwise be equal as MCSs, but are not equal as MCTs.

Balbiani and Vakarelov then claim that every consistent formula is satisfiable in *the* canonical $\mathrm{PDL}_0^{\cap,?}$ model (based on MCTs), c.f. [3, Prop. 6.3]. This is not true when taken literally, instead, their constructions prove the following weaker statement: every consistent $\mathrm{PDL}_0^{\cap,?}$ formula is satisfiable in *some* canonical model. This is a simple consequence of the fact that applications of rule (INT) introduce new atomic propositions that were not present in the language in the first place. In other words: the canonical model $\mathcal{K}$ whose worlds are MCTs depends on the underlying language.

Example 2. Consider $\mathrm{PDL}_0^{\cap,?}$ over the empty set of atomic propositions and let Φ_0 be the theory of the world with no successors. Clearly, Φ_0 is satisfiable and can therefore not be inconsistent with respect to a sound axiomatisation. As argued above, $\mathit{Split}(\Phi_0)$ is also satisfiable but its models must contain two disjoint copies of models of Φ_0. Given that Φ_0 is maximal, i.e. an MCS, and that rule (INT) is not applicable when no propositions are available and therefore every MCS is already an MCT we get that $\mathit{Split}(\Phi_0)$ is *not* satisfiable in *the* canonical for $\mathrm{PDL}_0^{\cap,?}$ over the empty set of atomic propositions. This shows that Prop. 6.3 of [3] needs to be taken with care, namely in the weaker sense stated above. Note that $\mathit{Split}(\Phi_0)$ is logically equivalent to a finite set of $\mathrm{PDL}_0^{\cap,?}$ formulae, and, hence, to a single formula.

Example 2 can be extended to any given set of propositions. Let Φ be any propositional labelling of the world with no successors that is complete in the sense that for every proposition p of the underlying $\mathcal{P}$ we have $p \in \Phi$ iff $\neg p \notin \Phi$. Then $\mathit{Split}(\Phi)$ requires two copies of this world to exist. Any invocation of the rule (INT) would require an *additional* proposition to distinguish the two. Consequently, the calculus of [3] considers the logic $\mathrm{PDL}_0^{\cap,?}$ in the language of some vocabulary τ but makes use of formulae that belong to the language of $\mathrm{PDL}_0^{\cap,?}$ in the vocabulary of a genuine superset τ^* of τ. It does not help to consider the larger vocabulary τ^* in the first place as the characterisation of the intersection operator uses propositions for every subset of a model, c.f. [3, Prop. 2.1]. This is clearly problematic in a canonical model construction when the propositions that are used to form the worlds are derived from the set of all subsets of worlds.

Balbiani has refined the construction of [3] in order to get rid of the need to introduce new propositions [1]. Weak completeness is proved using a similar canonical model construction, strong completeness is not achieved.

Finally, Balbiani and Vakarelov have also extended their work on $\mathrm{PDL}_0^{\cap,?}$ to the full $\mathrm{PDL}^{\cap,?}$ [4] also using a very similar canonical model construction based on a rule which requires new atomic propositions, and therefore statements about these models need to be taken with similar care. Most importantly,

$$[\alpha]p \leftrightarrow \neg\langle\alpha\rangle\neg p \quad \texttt{(Dl)}$$
$$\langle p?\rangle q \leftrightarrow p \wedge q \quad \texttt{(?)}$$
$$\langle \alpha \cap p?\rangle q \leftrightarrow \langle \alpha^{\circlearrowleft}\rangle(p \wedge q) \quad \texttt{(T1)}$$
$$[\alpha;\beta]p \leftrightarrow [\alpha][\beta]p \quad \texttt{(;)}$$
$$\langle \alpha \cup \beta\rangle p \leftrightarrow \langle\alpha\rangle p \vee \langle\beta\rangle p \quad \texttt{(D)}$$
$$\alpha \cap \beta \Rightarrow \alpha \quad \texttt{(Wk)}$$
$$\alpha \cap \beta \Leftrightarrow \beta \cap \alpha \quad \texttt{(Cm)}$$
$$\alpha \cap \alpha \Leftrightarrow \alpha \quad \texttt{(Ct)}$$
$$(\alpha \cup \beta);\gamma \Leftrightarrow (\alpha;\gamma) \cup (\beta;\gamma) \quad \texttt{(D3)}$$
$$\alpha;(\beta \cup \gamma) \Leftrightarrow (\alpha;\beta) \cup (\alpha;\gamma) \quad \texttt{(D4)}$$
$$\alpha \cap p? \Leftrightarrow (\langle \alpha^{\circlearrowleft}\rangle p)? \quad \texttt{(T)}$$

$$[\alpha](p \to q) \to [\alpha]p \to [\alpha]q \quad \texttt{(K)}$$
$$\langle \alpha^{\circlearrowleft}\rangle p \wedge \langle \beta^{\circlearrowleft}\rangle q \to \langle (\alpha;\beta)^{\circlearrowleft}\rangle(p \wedge q) \quad \texttt{(C1)}$$
$$[\alpha^{\circlearrowleft}]p \wedge [\beta^{\circlearrowleft}]p \to [\alpha^{\circlearrowleft};\beta^{\circlearrowleft}]p \quad \texttt{(C2)}$$
$$\langle \alpha^{\circlearrowleft}\rangle p \wedge [\alpha^{\circlearrowleft}]q \to p \wedge q \quad \texttt{(C3)}$$
$$\langle \alpha;(p \vee q)?;\beta\rangle r \leftrightarrow \langle \alpha;p?;\beta\rangle r \vee \langle \alpha;q?;\beta\rangle r \quad \texttt{(V)}$$
$$\alpha \cap (\beta \cap \gamma) \Leftrightarrow (\alpha \cap \beta) \cap \gamma \quad \texttt{(A)}$$
$$(\alpha;p?) \cap \beta \Leftrightarrow (\alpha \cap \beta);p? \quad \texttt{(T2)}$$
$$(p?;\alpha) \cap \beta) \Leftrightarrow p?;(\alpha \cap \beta) \quad \texttt{(T3)}$$
$$\alpha \cap (\beta \cup \gamma) \Leftrightarrow (\alpha \cap \beta) \cup (\alpha \cap \gamma) \quad \texttt{(D1)}$$
$$\alpha \cup (\beta \cap \gamma) \Leftrightarrow (\alpha \cup \beta) \cap (\alpha \cup \gamma) \quad \texttt{(D2)}$$
$$(\varphi \leftrightarrow \psi) \leftrightarrow (\varphi? \Leftrightarrow \psi?) \quad \texttt{(TP)}$$

$$\alpha^{\circlearrowleft} \Rightarrow \alpha\lceil(\beta_2;\beta_3;[(\beta_1;\beta_2;\beta_3)^{\circlearrowleft}]p?/\beta_2;[(\beta_3;\beta_1;\beta_2)^{\circlearrowleft}]p?;\beta_3\rfloor^{\circlearrowleft} \quad \texttt{(C)}$$

Figure 1: The formula and program axioms for $\mathrm{PDL}_0^{\cap,?}$.

there is no unique canonical model for all of $\mathrm{PDL}^{\cap,?}$ because its structure depends on the vocabulary of the underlying logic but the correctness proofs require its MCTs to be built using propositions for every subset of the model. At last, possible problems with canonical models for non-compact logics are avoided using an infinitary rule to handle Kleene stars in programs.

The following sections are devoted to the presentation of a sound and complete axiomatisation for $\mathrm{PDL}_0^{\cap,?}$ over an arbitrary vocabulary that works in the very same vocabulary.

4 Axiomatising $\mathrm{PDL}_0^{\cap,?}$

In this section we propose an axiomatisation for $\mathrm{PDL}_0^{\cap,?}$ and derive a normal form such that every $\mathrm{PDL}_0^{\cap,?}$ formula is equivalent to one in normal form, and this equivalence is also provable in the calculus.

4.1 Axioms and Rules

Let Δ be the smallest proof calculus that contains all propositional tautologies, the formula axiom schemes and program axioms schemes shown in Fig. 1 and the inference rules

$$\texttt{(MP)}\ \frac{\varphi \qquad \varphi \to \psi}{\psi} \qquad \texttt{(Gen)}\ \frac{\varphi}{[\alpha]\varphi} \qquad \texttt{(USub)}\ \frac{\varphi}{\varphi\lceil\psi/p\rfloor} \qquad \texttt{(PSub)}\ \frac{\varphi \qquad \alpha \Rightarrow \alpha'}{\varphi\lceil\langle\alpha'\rangle/\langle\alpha\rangle\rfloor}$$

where $\varphi\lceil\psi/p\rfloor$ is the usual substitution and $\varphi\lceil\langle\alpha'\rangle/\langle\alpha\rangle\rfloor$ is meant to denote that every program $\langle\alpha\rangle$ which occurs under an even number of negation symbols in the syntax tree of the formula is being replaced by $\langle\alpha'\rangle$. We use $\lceil\ \rfloor$ instead of the usual brackets for the substitution operator to distinguish them from the box modality.

We write $\vdash \varphi$ if the $\mathrm{PDL}_0^{\cap,?}$-formula φ can be derived from the axioms of Δ alone by repeated application of the rules of inference. For a set Φ of $\mathrm{PDL}_0^{\cap,?}$-formulae we write $\Phi \vdash \varphi$ if there exist $\varphi_1, \ldots, \varphi_n \in \Phi$ such that $\vdash (\bigwedge_{i=1}^n \varphi_i) \to \varphi$.

The purpose of rule (C) is to deal with properties of cyclic structures.

Example 3. Consider the formula $\varphi = \langle (a;\psi?;b)^{\circlearrowleft} \rangle \top$ which is satisfied at a state u which is the beginning of an $(a;b)$-cycle such that ψ is satisfied at an intermediate state v with $u \xrightarrow{a} v$ and $v \xrightarrow{b} u$. The satisfiability of φ depends on ψ, not just on whether it is satisfiable itself but also whether it is compatible with being satisfied on a cyclic structure.

Consider $\psi = [(b;a)^{\circlearrowleft}]\bot$. Clearly, ψ can not be satisfied on a state v with $v \xrightarrow{b} u$ and $u \xrightarrow{a} v$ for some u whence φ is not satisfiable at all. Rule C incorporates this into the calculus: Considering φ with $\psi = [(b;a)^{\circlearrowleft}]\bot$, a combination of rule C and modus ponens yields that $\langle (a;b;\psi'?)^{\circlearrowleft} \rangle \top$ with $\psi' = [(a;b)^{\circlearrowleft}]\bot$ is a logical consequence of φ. Using axiom T2 we also obtain $\langle (a;b)^{\circlearrowleft} \rangle \psi'$ with ψ' as before as a logical consequence. Using axiom C3 with $q = \top$ we obtain that ψ' is a also a logical conseqence of φ which makes φ inconsistent after a few derivations, correctly reflecting unsatisfiability of φ.

The intuition behind rule C is that it allows tests on cyclic programs to be transferred further along the cycle while correctly adjusting programs in these tests for the fact that they have been transferred *and* accounting for the fact that all this occurs on a cycle.

Lemma 2. *$\vdash$ is sound, i.e., $\Phi \vdash \varphi$ only if $\Phi \models \varphi$.*

The proof is by standard induction on the length of a proof. The rest of the paper is devoted to showing completeness of $\vdash$ using the notion of an MCS, c.f. Sect. 3. By Zorn's Lemma, every consistent formula set is contained in an MCS. Moreover, if Φ is an MCS, then it has the following properties: (1) Φ is closed under $\vdash$; (2) $\varphi \wedge \psi \in \Phi$ iff $\varphi \in \Phi$ and $\psi \in \Phi$; (3) $\varphi \vee \psi \in \Phi$ iff $\varphi \in \Phi$ or $\psi \in \Phi$; and (4) $\varphi \in \Phi$ iff $\neg\varphi \notin \Phi$ for any formula φ.

Lemma 3. *Let Φ, Ψ be* MCS*, X be a consistent set that is closed under $\vdash$ of $\mathrm{PDL}_0^{\cap,?}$-formulae and α_1 and α_2 be $\mathrm{PDL}_0^{\cap,?}$-programs, $\chi \in X$ such that $\langle \alpha_1;\chi?;\alpha_2 \rangle \psi \in \Phi$ for all $\psi \in \Psi$. Then the set $X' \coloneqq X \cup \{\varphi \mid [\alpha_1]\varphi \in \Phi\} \cup \{\langle \alpha_2 \rangle \psi \mid \psi \in \Psi\}$ is consistent.*

Proof. Assume X' was not consistent. Then, w.l.o.g., there are φ, ψ, χ such that $\chi \in X$, $[\alpha_1]\varphi \in \Phi$ and $\psi \in \Psi$, but $\vdash \varphi \wedge \chi \wedge \langle \alpha_2 \rangle \psi \to \bot$. We have $\langle \alpha_1;\chi?;\alpha_2 \rangle \psi \in \Phi$ and, hence, $\langle \alpha_1 \rangle (\chi \wedge \langle \alpha_2 \rangle \psi) \in \Phi$ as well as $[\alpha_1]\varphi \in \Phi$, so $\langle \alpha_1 \rangle (\varphi \wedge \chi \wedge \langle \alpha_2 \rangle \psi) \in \Phi$ which contradicts the assumption. □

Lemma 4. *Let Φ, Ψ, Υ be* MCS*, X be consistent and closed under $\vdash$, and $\alpha_1, \alpha_2, \beta_1, \beta_2$ be programs. Let*

1. *$\langle \alpha_1;\chi?;\alpha_2 \rangle \psi \in \Phi$ for all $\psi \in \Psi$, $\chi \in X$,*
2. *$\langle \big((\beta_2;\upsilon?;\beta_1);\varphi?;(\alpha_1;\chi?;\alpha_2)\big)^{\circlearrowleft} \rangle \top \in \Psi$ for all $\upsilon \in \Upsilon, \varphi \in \Phi, \chi \in X$,*
3. *$\langle \big((\alpha_1;\chi?;\alpha_2);\psi?;(\beta_2;\upsilon?;\beta_1)\big)^{\circlearrowleft} \rangle \top \in \Phi$ for all $\chi \in X, \psi \in \Psi, \upsilon \in \Upsilon$,*
4. *$\langle ((\beta_1;\varphi?;\alpha_1;);\chi?;(\alpha_2;\psi?;\beta_2))^{\circlearrowleft} \rangle \top \in \Upsilon$ for all $\varphi \in \Phi, \chi \in X, \psi \in \Psi$.*

Then the following set X^* is consistent.

$$\begin{aligned} X^* \;=\; & X \cup \{\varphi \mid [\alpha_1]\varphi \in \Phi\} \cup \{\langle \alpha_2 \rangle \psi \mid \psi \in \Psi\} \\ & \cup \{\langle \big((\alpha_2;\psi?;\beta_2);\upsilon?;(\beta_1;\varphi?;\alpha_1)\big)^{\circlearrowleft} \rangle \top \mid \psi \in \Psi, \upsilon \in \Upsilon, \varphi \in \Phi\} \end{aligned}$$

Proof. We show that the union of X and the third set is consistent. Assume that it is not. Then there are finitely many formulae $\zeta_1, \ldots, \zeta_n$ of the form $\zeta_i = \langle((\alpha_2;\psi?;\beta_2);\upsilon?;(\beta_1;\varphi?;\alpha_1))^{\circlearrowleft}\rangle\top$, with programs and formulae as suggested such that $\zeta = \bigvee_{i=1}^{n}\neg\zeta_i \in X$. Then $\langle(((\beta_1;\varphi?;\alpha_1;);\zeta?;(\alpha_2;\psi?;\beta_2))^{\circlearrowleft}\rangle\top \in \Upsilon$ and, since Υ is an MCS, also rule (V) implies that also $\langle(((\beta_1;\varphi?;\alpha_1;);\neg\zeta_i?;(\alpha_2;\psi?;\beta_2))^{\circlearrowleft}\rangle\top \in \Upsilon$ for at least one i. Now we can apply rule (C) to conclude that $\langle(((\beta_1;\varphi?;\alpha_1;);\top?;(\alpha_2;\psi?;\beta_2);\neg\zeta_i'))^{\circlearrowleft}\rangle\top \in \Upsilon$ where $\neg\zeta_i'$ is equivalent to $[((\beta_1;\varphi?;\alpha_1);\top?;(\alpha_2;\psi?;\beta_2);\upsilon?)^{\circlearrowleft}]\bot$ which contradicts consistency of Υ since then also $\neg\zeta_i' \in \Upsilon$.

Similar arguments show that we can also add the other two sets without losing consistency. □

4.2 A Normal-Form Lemma for PDL$_0^{\cap,?}$

We partition the PDL$_0^{\cap,?}$-programs into two groups. The first one consists of *cyclic* programs, *Cyc* in short. They test if something holds at the present state, possibly requiring this state to have a (perhaps complex) self-loop. Thus, they force the evaluation of a formula to stay at the current node. Secondly, there are the *Forw*-programs which make up all others. These require the evaluation of a formula to take at least one step into some direction. Syntactically, cyclic and forward programs are defined as follows:

$$\alpha_{\mathsf{forw}} \;\coloneqq\; a \mid \alpha_{\mathsf{forw}} \cap \alpha_{\mathsf{forw}} \mid \alpha_{\mathsf{forw}};\alpha_{\mathsf{cyc}};\alpha_{\mathsf{forw}} \qquad\qquad \alpha_{\mathsf{cyc}} \;\coloneqq\; \varphi? \mid \alpha_{\mathsf{forw}} \cap \varphi?$$

Lemma 5. *For every φ there is a φ' such that $\vdash \varphi \leftrightarrow \varphi'$ and all programs in φ' belong to $Cyc \cup Forw$. Moreover, for all formulae of the form $\langle\alpha_{\mathsf{cyc}}\rangle\varphi$ we have $\vdash \langle\alpha_{\mathsf{cyc}}\rangle\varphi \leftrightarrow \langle\alpha_{\mathsf{forw}}^{\circlearrowleft}\rangle\varphi'$ for some formula φ' or just $\vdash \langle\alpha_{\mathsf{cyc}}\rangle\varphi \leftrightarrow \varphi'$ for some formula φ' without the modal prefix $\langle\alpha_{\mathsf{cyc}}\rangle$.*

Proof. We will show this in several steps. First, we eliminate the disjunction operator from programs. Using (Cm), (D1),...,(D4), we can transform every program into one in which $\cup$ does not occur underneath a different program operator. Using axiom (D) it is possible to eliminate occurrences of $\cup$ in such top-level positions in programs. In the following, α_{cyc} denotes an arbitrary program from *Cyc*.

Next we note that *Cyc*-programs commute with the intersection operator if they are at the end or the beginning of a sequential composition: Using axiom (T) and some basic propositional logic, one can check that $\vdash \alpha \cap \psi? \;\Leftrightarrow\; (\langle\alpha^{\circlearrowleft}\rangle\top \wedge \psi)?$ and thus every *Cyc*-program is equivalent to a test. Consequently with (T2), (T3) we can derive that

$$\vdash (\alpha_{\mathsf{cyc}};\alpha) \cap \alpha' \Leftrightarrow \alpha_{\mathsf{cyc}};(\alpha \cap \alpha') \text{ and } \vdash (\alpha;\alpha_{\mathsf{cyc}}) \cap \alpha' \Leftrightarrow (\alpha \cap \alpha');\alpha_{\mathsf{cyc}}.$$

Note that after using this equivalence α_{cyc} is at the beginning or the end of the respective sequence. We can thus assume that every occurence of a *Cyc*-program is as high as possible in the syntax-tree of a program.

In the next step we want to eliminate, resp. simplify isolated *Cyc*-programs, i.e. programs α_{cyc} in a formula $\langle\alpha_{\mathsf{cyc}}\rangle\varphi$. Using (T1) we get

$$\vdash \langle\psi?\rangle\varphi \Leftrightarrow \varphi \wedge \psi \qquad \text{and} \qquad \vdash \langle\psi? \cap \alpha\rangle\varphi \Leftrightarrow \langle\alpha \cap \top?\rangle(\varphi \wedge \psi)\,.$$

Note that these equivalences entail the second and third statement of the lemma.

Similarly we can simplify *Cyc*-programs at the beginning or end of a program that is just a sequential composition by first using (;) and then (T1) again.

We have just dealt with *Cyc*-programs at the beginning or the end of a sequence of sequential compositions and moved them up in the syntax tree wherever possible. It remains to be seen how they are treated in the middle of such a sequence.

Let $\alpha;\alpha_{\mathsf{cyc}};\alpha'$ be a program with a cyclic program in the middle of a sequence. We make a case distinction over α. The part for α' is symmetric. In case α is atomic or the intersection of *Forw*-programs, we are finished. If $\alpha = \beta;\beta_{\mathsf{cyc}}$, then $\alpha;\alpha_{\mathsf{cyc}};\alpha' = \beta;\beta_{\mathsf{cyc}};\alpha_{\mathsf{cyc}};\alpha'$. Again, we use the fact that each *Cyc*-program is equivalent to a simple test. Using $(;)$ and $(?)$ several times we get $\vdash \beta;\beta_{\mathsf{cyc}};\alpha_{\mathsf{cyc}};\alpha' \Leftrightarrow \beta;\beta_{\mathsf{cyc}} \cap \alpha_{\mathsf{cyc}};\alpha$. Note, that *Cyc*-programs are closed under intersections.

The last case is that of $\alpha = \beta;\beta'$ where both are simple *Forw*-programs. Using basic propositional logic, $(;)$ and $(?)$ we get $\vdash \beta;\beta' \Leftrightarrow \beta;\top?;\beta'$.

□

As a consequence of the proof of this lemma, we will sometimes assume w.l.o.g. that cyc-programs are of the form $\varphi?$ for some suitable φ.

5 A Canonical Model for $\mathrm{PDL}_0^{\cap,?}$

5.1 Large Programs

In order to maintain induction invariants during the construction of our canonical model, we need to allow tests to test against arbitrary sets of formulae as opposed to just one formula. We call the resulting extension of programs *large programs*. This is not exactly the same notion of large programs as in [4].

Definition 6. The set of *large programs* is defined inductively via

$$\alpha ::= a \mid \alpha \cap \alpha \mid \alpha;\Phi?,\alpha$$

where Φ is a consistent set of $\mathrm{PDL}_0^{\cap,?}$-formulae. A large loop is of the form $\alpha^{\circlearrowleft}$, where α is a large program.

An ordinary program α is an *instance* of a large program α_l if

- $\alpha = \alpha_l = a$ for some accessibility relation a or,
- $\alpha = \alpha^1 \cap \alpha^2, \alpha^l = \alpha_l^1 \cap \alpha_l^2$ and α^i is an instance of α_l^i for $i = 1,2$ or,
- $\alpha = \alpha^1;\varphi?;\alpha^2, \alpha_l = \alpha_l^1;\Phi?;\alpha_l^2$, $\varphi \in \Phi$ and α^i is an instance of α_l^i for $i = 1,2$.

A loop $\alpha^{\circlearrowleft}$ is an instance of a large loop $\alpha_l^{\circlearrowleft}$ if α is an instance of α_l.

We write $\alpha \leq \alpha'$ for large programs α, α' if and only if every instance of α is an instance of α'.

A large program is an ordinary program where tests against a formula have been replaced with tests against a set of formulae. Clearly, an ordinary program with consistent tests can be made large by replacing tests of the form $\varphi?$ by tests of the form $\{\varphi\}?$, and the original program will be an instance of the new large program.

Definition 7. Let Φ, Ψ be MCS and let α be a large program. For occurrences of subprograms β define the left and right sets $l(\beta)$ and $r(\beta)$ in a top-down manner via

- $l(\alpha) := \Phi, r(\alpha) := \Psi$,
- If $\alpha = \alpha_1 \cap \alpha_2$ then $l(\alpha_1) = l(\alpha_2) := l(\alpha)$ and $r(\alpha_1) = r(\alpha_2) := r(\alpha)$,
- If $\alpha = \alpha_1;X?;\alpha_2$ then $l(\alpha_1) := l(\alpha), r(\alpha_1) = l(\alpha_2) := X, r(\alpha_2) := r(\alpha)$.

In the case of a large loop $\alpha^{\circlearrowleft}$ and a set Φ, define the left and right programs $lp(X)$ and $rp(X)$ of an occurrence of a test X in a top-down manner as follows:

- $lp(\Phi) \coloneqq \top?, rp(\Phi) \coloneqq \top?$,
- If $\alpha = \alpha_1; X?; \alpha_2$ then $lp(X) \coloneqq lp(l(\alpha)); l(\alpha)?; \alpha_1$ and $rp(X) \coloneqq \alpha_2; r(\alpha)?, rp(r(\alpha))$.

We say that $\Phi \xrightarrow{\alpha} \Psi$ is *consistent* if $[\beta']\neg\psi \notin l(\beta)$ for all instances β' of subprograms β of α and all $\psi \in r(\beta)$, and all test sets are consistent. We say that $\Phi \xrightarrow{\alpha} \Psi$ is inconsistent if it is not consistent. A large loop $\alpha^{\circlearrowleft}$ is consistent at Φ if the above holds with $\Phi = \Psi$ and for each test set X, no formula of the form $[(\beta_1; \varphi?; \beta_2)^{\circlearrowleft}]\bot$, with β_1 an instance of $rp(X)$, β_2 an instance of $lp(X)$ and $\varphi \in \Phi$ is in X.

We say that $\Phi \xrightarrow{\alpha} \Psi$ is *maximally consistent* if $\Phi \xrightarrow{\alpha} \Psi$ is consistent and every test in α is an MCS. In particular, for all subprograms $\beta = \beta_1; X?; \beta_2$ we have that $X \supseteq \{\varphi \mid [\beta_1']\varphi \in l(\beta), \beta_1' \text{ instance of } \beta_1\} \cup \{\langle\beta_2'\rangle\psi \mid \psi \in r(\beta), \beta_2' \text{ instance of } \beta_2\}$. We say that a large loop $\alpha^{\circlearrowleft}$ is maximally consistent at Φ if $\Phi \xrightarrow{\alpha} \Phi$ is maximally consistent and, additionally, for all tests $X?$,

$$\{\langle(\beta_1; \varphi?\beta_2)^{\circlearrowleft}\rangle\top \mid \beta_1 \text{ instance of } lp(X), \beta_2 \text{ instance of } rp(X), \varphi \in \Phi\} \subseteq X.$$

Lemma 8. *Let $\Phi \xrightarrow{\alpha} \Psi$ for a large program α be consistent. Then there is a large program $\alpha' \geq \alpha$ such that $\Phi \xrightarrow{\alpha'} \Psi$ is maximally consistent. Moreover, if $\alpha^{\circlearrowleft}$ is a large loop such that $\Phi \xrightarrow{\alpha^{\circlearrowleft}} \Phi$ is consistent, there is $\alpha' \geq \alpha$ such that $\Phi \xrightarrow{\alpha'} \Phi$ is maximally consistent.*

Proof. Let Ψ, Φ and α be as in the lemma. For the case of a large loop, set $\Psi = \Phi$. For convenience, we assume that the test sets in every subprogram of α are closed under conjunctions, if not, we close them under conjunctions. Clearly this will not make $\Phi \xrightarrow{\alpha} \Psi$ inconsistent.

The proof proceeds recursively: Assume that β is a subprogram of α such that $l(\beta)$ and $r(\beta)$ are already MCS. There are three cases: $\beta = a$ for some atomic program a, $\beta = \beta_1 \cap \beta_2$ and $\beta = \beta_1; X?, \beta_2$. In the first case, there is nothing left to do. In the second case, convert all the test sets in β_1 into MCS such that $\Phi \xrightarrow{\alpha} \Psi$ stays consistent. Then repeat the same procedure with β_2.

In the third case, we have to find an MCS X^* that is a superset of X such that replacing X by X^* will not break consistency of $\Phi \xrightarrow{\alpha} \Psi$. W.l.o.g., X is already closed under $\vdash$. By Lemma 3, the set $X' = X \cup \{\varphi \mid [\beta_1]\varphi \in l(\beta)\} \cup \{\langle\beta_2\rangle\psi \mid \psi \in r(\beta)\}$ is consistent. Moreover, replacing X by X' will not make $\Phi \xrightarrow{\alpha} \Psi$ inconsistent because otherwise, Φ would be inconsistent. By Lemma 4, in the case of a large loop, set X' as

$$\begin{aligned} &X \cup \{\varphi \mid [\beta_1]\varphi \in l(\beta)\} \\ &\quad \cup \{\langle\beta_2\rangle\psi \mid \psi \in r(\beta)\} \\ &\quad \cup \{\langle((\beta_2; \psi?; rp(r(\beta))); \upsilon?; (lp(l(\beta)); \varphi?; \beta_1))^{\circlearrowleft}\rangle\top \mid \psi \in r(\beta), \upsilon \in \Phi, \varphi \in l(\beta)\} \end{aligned}$$

which is also consistent. Now consider the set $\mathcal{X}$ of all consistent supersets of X' such that $\Phi \xrightarrow{\alpha} \Psi$ stays consistent if X is replaced by a set from $\mathcal{X}$. This set is nonempty, because it contains X'. It is also partially ordered by set inclusion and the union of any chain of sets from $\mathcal{X}$ is in $\mathcal{X}$ for otherwise, there would be a minimal set in the chain which also is not in $\mathcal{X}$.

By Zorn's Lemma, $\mathcal{X}$ contains a maximal element X^*. We argue that X^* is an MCS. Assume otherwise, then there is $\varphi \in \text{PDL}_0^{\cap,?}$ such that neither φ nor $\neg\varphi$ are in X^*. Since X^* is consistent, one of φ and $\neg\varphi$ can be added to X^* without losing consistency of X^*. Hence, by assumption, replacing X in β by either of $X \cup \{\varphi\}$ and $X \cup \{\neg\varphi\}$ will make $\Phi \xrightarrow{\alpha} \Psi$ inconsistent, but using X^* itself does not. Then there are instances α_1 and α_2 of α such that both $[\alpha_1\lceil(\beta_1; \varphi?; \beta_2)/\beta\rceil]\neg\psi_1$ and $[\alpha_2\lceil\beta_1; \neg\varphi?; \beta_2/\beta\rceil]\neg\psi_2$ are in Φ for $\psi_1, \psi_2 \in \Psi$. Since Ψ and all test sets are closed under conjunction, we can assume that $\alpha_1 = \alpha_2$ and $\psi_1 = \psi_2$. But $\vdash [\alpha_1\lceil(\beta_1; (\varphi \vee \neg\varphi)?; \beta_2)/\beta\rceil]\neg\psi_1 \leftrightarrow [\alpha_1]\neg\psi$ and $[\alpha_1]\neg\psi \in \Phi$, which is a contradiction to

X^* being a safe replacement for X. This contradiction stems from the assumption that X^* is not maximal. Hence, X^* is the desired MCS. The process continues recursively with β_1 and β_2.

It remains to argue that for all subprograms $\beta = \beta_1;X?;\beta_2$ we have that the set inclusion $X \supseteq \{\varphi \mid [\beta_1']\varphi \in l(\beta), \beta_1' \text{ instance of } \beta_1\} \cup \{\langle\beta_2'\rangle\psi \mid \psi \in r(\beta), \beta_2' \text{ instance of } \beta_2\}$ holds. For the first component, this is because we made sure the relevant φ are in X before proceeding to make the tests in β_1 maximal. Since $\vdash [\alpha_1;\Phi?;\alpha_2]\psi$ entails $[\alpha_1;\Phi'?;\alpha_2]\psi$ for $\Phi' \supseteq \Phi$, there is nothing left to prove. For the second part, assume that there is an instance β_2' of β_2 such that $\langle\beta_2'\rangle\psi \notin X$ for some $\psi \in r(\beta)$. Since X is an MCS, $[\beta_2']\neg\psi \in X$. But then $\Phi \xrightarrow{\alpha} \Psi$ is not even consistent, which contradicts the fact that all of the induction steps maintain consistency. In the case of a large loop, the same argument entails that all formulae required for a large loop to be maximally consistent are present at the tests. □

5.2 Construction of the Canonical Model

For the rest of this section, we assume, that all formulae are in the form defined in Lemma 5. Let τ' be τ extended with accessibility relation symbols α and α' for all *large* $\mathrm{PDL}_0^{\cap,?}[\tau]$-programs α. We use α-edges for *Forw*-programs and α'-edges for *Cyc*-programs to better differentiate the two types and refer to the former as abstract forward edges and the latter as abstract loop edges.

Each point in the canonical model we construct is labeled by an $\mathrm{PDL}_0^{\cap,?}[\tau]$-MCS such that, after the construction is complete, a formula holds at a point if and only if it is in the MCS that labels that point. The construction proceeds inductively. Each induction step consists of two stages: In the first stage, we add new points that witness the truth of diamond-type formulae $\langle\alpha\rangle\varphi$ at nodes of the previous induction step. These new states are connected to the previous ones via abstract β-edges from $\tau' \setminus \tau$, where β is a suitable large program derived from α. In the second stage these abstract β-edges are converted into subgraphs such that if there is an abstract β-edge from a node u to a node v, then u is connected via α to v. This is done by adding edges for abstract subprograms of β and intermediate points, if necessary. The subgraph created for this is called the *arena* witnessing that α connects u to v. The whole process proceeds in a fashion such that no box-type formulae of the form $[\alpha]\varphi$ are violated.

The desired canonical model is the τ-reduct of the limit of the inductive process. During this process, many points are labeled by the same MCS. Since this can not be avoided (see Section 3), labels of points are more complex.

Let $\mathrm{mcs}(\mathrm{PDL}_0^{\cap,?})$ be the set of all maximally consistent $\mathrm{PDL}_0^{\cap,?}$-sets. Then

$$\begin{aligned}\mathrm{Gen}_0 &= \mathrm{mcs}(\mathrm{PDL}_0^{\cap,?})\\ \mathrm{Gen}_{i+1} &= \{\mathrm{arena}(\Psi,\alpha,l) \mid \Psi \in \mathrm{mcs}(\mathrm{PDL}_0^{\cap,?}), \alpha \in \mathit{Forw}, l \in \mathrm{Gen}_i\}\\ &\quad \cup \{\mathrm{arena}(\alpha^{\circlearrowleft},l) \mid \alpha \in \mathit{Forw}, l \in \mathrm{Gen}_i\}\end{aligned}$$

with $\mathrm{arena}(\Psi,\alpha,l)$, $\mathrm{arena}(\alpha,l)$ and $\mathrm{dom}(l)$ defined inductively as follows:

- If $l \in \mathrm{Gen}_0$ then $\mathrm{dom}(l) = l$.
- If not $\langle\alpha\rangle\psi \in \mathrm{dom}(l)$ for all $\psi \in \Psi$, then $\mathrm{arena}(\Psi,\alpha,l)$ is empty.
- If not $\langle\alpha^{\circlearrowleft}\rangle\top \in \mathrm{dom}(l)$ then $\mathrm{arena}(\alpha^{\circlearrowleft},l)$ is empty.
- If $\langle\alpha\rangle\varphi \in \mathrm{dom}(l)$ for all φ in some MCS Φ, then there is a large program α' such that $\Psi \xrightarrow{\alpha'} \Phi$ is maximally consistent. Then $\mathrm{arena}(\Phi,\alpha,l)$ is the nodes l, $r = (\Phi,\alpha,l)$ together with the abstract α'-forward edge from l to r and the subgraph induced by it. Moreover, $\mathrm{dom}(r) = \Phi$.

- If $\langle \alpha^{\circlearrowleft} \rangle \top \in \mathrm{dom}(l)$ then there is an abstract α'-loop such that $\Psi \xrightarrow{\alpha'^{\circlearrowleft}} \Psi$ is maximally consistent. Then $\mathrm{arena}(\alpha^{\circlearrowleft}, l)$ is l together with the abstract α'-loop edge and the subgraph induced by it.

The subgraphs induced by abstract forward edges are again defined inductively:

- The subgraph induced by an abstract forward edge or an abstract loop edge α with α of the form a for some atomic program $a \in \tau$ is an a-edge.
- The subgraph induced by an abstract forward edge of the form $\alpha = \alpha_1; X?; \alpha_2$ from u to v consists of a node w with $\mathrm{dom}(w) = X$, an abstract α_1-forward edge from u to w, an abstract α_2-forward edge from w to v and the subgraphs induced by the abstract α_i. Note that, by consistency of $u \xrightarrow{\alpha} v$, we have $X \supseteq \{\varphi \mid [\alpha_1']\varphi \in \mathrm{dom}(u)\} \cup \{\langle \alpha_2' \rangle \psi \mid \psi \in \mathrm{dom}(v)\}$ for all instances α_1' of α_1 and α_2' of α_2.
- The subgraph induced by an abstract forward edge of the form $\alpha = \alpha_1 \cap \alpha_2$ from u to v consists of abstract forward edges α_1 and α_2 from u to v and the subgraphs induced by them.

For abstract loop edges, the process is similar, but we annotate nodes with the programs needed to complete the loop in question and the set from which the loop starts. For an abstract α-loop edge at u set $l(u) = r(u) = \top?$.

- The subgraph induced by an abstract loop edge of the form $\alpha = \alpha_1; X?; \alpha_2$ from u to v, which is part of a loop at s, consists of a node $w = (lp(w), X, rp(w), s)$ with $\mathrm{dom}(w) = X$, $lp(w) = (lp(u); \mathrm{dom}(u)?; \alpha_1)$, $rp(w) = (\alpha_2; \mathrm{dom}(v)?; rp(v))$, an abstract α_1-loop edge from u to w, an abstract α_2-loop edge from w to v and the subgraphs induced by the abstract loop edges α_i. Note that, by consistency of $u \xrightarrow{\alpha} v$ and by Lemma 8, we have

$$\begin{aligned} X \supseteq & \{\varphi \mid [\alpha_1']\varphi \in \mathrm{dom}(u)\} \\ & \cup \{\langle \alpha_2' \rangle \psi \mid \psi \in \mathrm{dom}(w)\} \\ & \cup \{\langle \big((\alpha_2'; \psi?; \beta_2'); \upsilon?; (\beta_1'; \varphi?; \alpha_1) \big)^{\circlearrowleft} \rangle \top \mid \psi \in v, \upsilon \in s, \varphi \in u\} \end{aligned}$$

  for all instances $\alpha_1', \alpha_2', \beta_2, \beta_1$ of $\alpha_1, \alpha_2, l(v), r(v)$.
- The subgraph induced by an abstract loop edge of the form $\alpha = \alpha_1 \cap \alpha_2$ from u to v consists of abstract loop edges α_1 and α_2 from u to v and the subgraphs induced by them.

Definition 9. The canonical model for $\mathrm{PDL}_0^{\cap,?}[\tau]$ is the τ-reduct of the structure $\mathfrak{A} = \langle A, \{p^{\mathfrak{A}}\}_{p \in \mathcal{P}}, \{R^{\mathfrak{A}}\}_{R \in \mathcal{R}} \rangle$, such that $A = \bigcup_{i=0}^{\infty} \mathrm{Gen}_i$, $P^{\mathfrak{A}} = \{l \in A \mid P \in \mathrm{dom}(l)\}$ and $R^{\mathfrak{A}}$ as described above. The union is meant to be disjoint, with the exception that points $l \in \mathrm{Gen}_i$ and their counterparts in sets of the form $\mathrm{arena}(\Phi, \alpha, l)$ and $\mathrm{arena}(\alpha^{\circlearrowleft}, l)$ are identified.

Lemma 10. *1. If there is an abstract α-forward edge from u to v, then for all $[\alpha']\psi \in \mathrm{dom}(u)$ and α' an instance of α, we have $\psi \in \mathrm{dom}(v)$.*

2. The set of arenas in $\mathfrak{A}$ decomposes $\mathfrak{A}$ into a forest-like structure. Any two arenas share at most one node, and any path from one arena to the other must go trough that node if it exists. For disjoint arenas, there is a node so that any path from one arena to the other must go through that node.

3. Any cycle in $\mathfrak{A}$ consists purely of nodes induced by abstract loop edges. Any α-cycle at a node u that consists purely of loop edges from the same arena is such that $\langle \alpha'^{\circlearrowleft} \rangle \top \in \mathrm{dom}(u)$ for any instance α' of α.

The proof is immediate from the construction of the model.

5.3 Soundness and Completeness of the Canonical Model

Lemma 11 (Gateway Lemma)**.** *Let $\alpha \in Forw$ be a program which contains at least one occurrence of the ;-operator. Let $\mathfrak{A}$ be a Kripke structure and let u and w be not necessarily distinct worlds in $\mathfrak{A}$ such that $\mathfrak{A}$ is a minimal witness graph for $u \xrightarrow{\alpha} v$. Let v be a node different from u and w such that all paths of length > 0 from u to w must go through v. Then there is $\beta = \beta_1;\beta_2$ such that $\vdash \beta \Rightarrow \alpha$ and $u \xrightarrow{\beta_1} w$ and $w \xrightarrow{\beta_2} v$.*

Proof. The proof is by induction on the construction of α. Since α contains at least one occurrence of the ;-operator, $\alpha \neq \bigcap_{i\in I} R_i$ for a finite set of $R_i \in \mathcal{R}$. Hence, the base case is that $\alpha = \alpha_1;\alpha_2$ and $u \xrightarrow{\alpha_2} v$ and $v \xrightarrow{\alpha_2} w$.

If $\alpha = \alpha_1;\alpha_2$ and not $u \xrightarrow{\alpha_1} v$ or not $v \xrightarrow{\alpha_2} w$, then there is v' such that $u \xrightarrow{\alpha_1} v'$ and $v' \xrightarrow{\alpha_2} w$. There are two cases: All paths from u to v' go through v, or all paths from v' to w go through v. Otherwise, there is a path from u to w via v' that does not go through v. If all paths from u to v' go through v, by the induction hypothesis there is $\alpha_1';\alpha_1''$ such that $u \xrightarrow{\alpha_1'} v$ and $v \xrightarrow{\alpha_1''} v'$. Then $\beta_1 = \alpha_1'$ and $\beta_2 = \alpha_1'';\alpha_2$ are as desired. If all paths from v' to w go through v', an application of the induction hypothesis yields α_2' and α_2'' such that $u \xrightarrow{\alpha_1;\alpha_2'} v$ and $v \xrightarrow{\alpha_2''} w$. In both cases, clearly $\vdash [\alpha]p \Rightarrow [\beta]p$.

If $\alpha = \alpha_1 \cap \alpha_2$, then by the induction hypothesis there are $\alpha_1';\alpha_1''$ and $\alpha_2';\alpha_2''$ such that $u \xrightarrow{\alpha_i'} w$ and $v \xrightarrow{\alpha_i''} v$ for $i = 1,2$. Then, via Axiom (;), we have $\vdash (\alpha_1' \cap \alpha_2');(\alpha_1'' \cap \alpha_2'') \Rightarrow \alpha_1 \cap \alpha_2$.

□

Lemma 12. *Let α be a program. Let $\mathfrak{A}$ be a Kripke structure and let u and w be not necessarily distinct worlds in $\mathfrak{A}$ such that $\mathfrak{A}$ is a minimal witness graph for $u \xrightarrow{\alpha} w$. Let v be a node and E a nonempty set of edges such that all paths from u to w are such that v occurs before and after any edge $e \in E$.*

Then there are programs α' and β such that α' contains an occurrence of $\langle \beta^{\circlearrowleft} \rangle \top ?$, $\vdash \alpha' \Rightarrow \alpha$ and the witness graph for α'' contains no node from X.

Proof. By application of Lemma 11 and the fact that $\vdash \beta^{\circlearrowleft} \Rightarrow \beta$. □

Lemma 13. *Let $u,v \in A$ and α_1, α_2 be large programs. If there are abstract R_{α_1}- and abstract R_{α_2}-edges between u and v either both forward or loop edges, then either $\alpha_1 = \alpha_2$, or there is a large program α of the same kind such that $\alpha \geq \alpha_1$ and $\alpha \geq \alpha_2$, and there is an abstract R_α-edge between u and v.*

Proof. By the construction of arenas. Each abstract edge α that is not that inducing an arena itself has a parent edge α' in the inductive process such that the top operator of α' is either ; or $\cap$. Usage of the ;-operator changes either source or sink of an abstract edge, and for different programs intermediate points are different. Since α_1 and α_2 share the same source and sink nodes, they must have a common parent solely via $\cap$-operators. □

Lemma 14. *Let $u,v,w \in A$ and α_1, α_2 be large programs. If there is an abstract α_1-edge from u to w and an abstract α_2-edge from w to v, then either there is an abstract α-edge from u to v and $\alpha \geq \alpha_1;\alpha_2$, or all paths from u to v contain w.*

Proof. By construction of arenas. If u and v are in different arenas the claim follows from Item (2) of Lemma 10. Otherwise, each abstract edge α that is not inducing an arena itself has a parent edge α' in the inductive process such that the top operator of α' is either ; or $\cap$. If there is no common parent of

α_1 and α_2 from u to v, then at least one of the α_i must have a program with top operator ; as its most common parent. By the construction of arenas, all paths from u to v must go through w. □

Lemma 15. *For all points $u,v \in A$ with and all programs α such that $u \xrightarrow{\alpha} v$ and all formulae ψ it holds that: If $[\alpha]\psi \in dom(u)$, then $\psi \in dom(v)$.*

Proof. If $u = v$, because $\vdash [\alpha]\psi \to [\alpha^\circlearrowleft]\psi$, we can invoke Lemma 16. So without loss of generality, $u \neq v$.

Since $u \neq v$, we know that $\alpha \in Forw$. Moreover, we can assume that both u and v and the witness graph for $u \xrightarrow{\alpha} v$ are both contained in the same arena. Otherwise, u and v are in different arenas and, by Item (2) of Lemma 10, there is w such that all paths from u to v must go through w. By the Gateway Lemma 11, α can be rewritten as $\alpha_1;\alpha_2$ such that $u \xrightarrow{\alpha_1} w$ and $w \xrightarrow{\alpha_2} v$. The claim of the lemma reduces to prove that $\psi' = [\alpha_2]\psi \in \mathrm{dom}(w)$. In a similar fashion, if u and v are in the same arena, but parts of the witness graph of $u \xrightarrow{\alpha} v$ are not, the conditions of Lemma 12 are met and parts in different arenas can be reduced to a test.

We prove that there is a sequence $\beta_1;\ldots;\beta_n$ of abstract edges such that there are instances β_i' of the β_i with $\vdash \beta_1';\ldots;\beta_n' \Rightarrow \alpha$. By Item (1) of Lemma 10, this proves the lemma. We prove this by induction over the structure of α. If $\alpha = a$ for some atomic program a, then by construction, there is an abstract a-edge from u to v. If $\alpha = \alpha_1;\alpha_2$, there is w such that $u \xrightarrow{\alpha_1} w$ and $w \xrightarrow{\alpha_2} v$. By the induction hypothesis there are sequences $\beta_1^1;\ldots;\beta_n^1$ and $\beta_1^2;\ldots;\beta_m^2$ and points $u = w_0^1,\ldots,w_n^1 = w$ and $w = w_0^2;\ldots;w_m^2 = v$ such that there are abstract β_i^j-edges from w_{i-1}^j to w_i^j and there are instances β_i^j of the β_i^j such that $\vdash \beta_1^1;\ldots;\beta_n^1 \Rightarrow \alpha_1$ and $\vdash \beta_1^2;\ldots;\beta_m^2 \Rightarrow \alpha_2$. Then $\vdash \beta_1^1;\ldots;\beta_n^1;\beta_1^2;\ldots;\beta_m^2 \Rightarrow \alpha_1;\alpha_2$.

If $\alpha = \alpha_1 \cap \alpha_2$, then $u \xrightarrow{\alpha_1} v$ and $u \xrightarrow{\alpha_2} v$. By the induction hypothesis, there are sequences $\beta_1^1;\ldots;\beta_n^1$ and $\beta_1^2;\ldots;\beta_m^2$ and points $u = w_0^1,\ldots,w_n^1 = v$ and $u = w_0^2;\ldots;w_m^2 = v$ such that there are abstract β_i^j-edges from w_{i-1}^j to w_i^j and there are instances β_i^j of the β_i^j such that $\vdash \beta_1^1;\ldots;\beta_n^1 \Rightarrow \alpha_1$ and $\vdash \beta_1^2;\ldots;\beta_m^2 \Rightarrow \alpha_2$. There are two cases: If both sequences have length one, we can apply Lemma 13 to obtain an abstract edge β from u to v such that there are instances β_1 and β_2 of β with $\vdash \beta_1 \Rightarrow \alpha_1$ and $\vdash \beta_2 \Rightarrow \alpha_2$. Then the sequence just consisting of β is as desired.

If at least one sequence has length longer than one, we begin replacing subsequences of the form $\beta_i^j;\beta_{i+1}^j$ by abstract $\beta_i^j;\beta_{i+1}^j$-edges, if possible. Either both sequences reach length one, and we can apply the previous case, or the conditions of Lemma 14 apply and there is w such that all paths from u to v go through w. By the Gateway Lemma 11, we can rewrite α_1 and α_2 to $\alpha_1^1;\alpha_1^2$ and $\alpha_2^1;\alpha_2^2$ such that $\vdash \alpha_i^1;\alpha_i^2 \Rightarrow \alpha_i$ and $u \xrightarrow{\alpha_i^1} w$ and $w \xrightarrow{\alpha_i^2} v$ for $i = 1,2$. Then $\vdash (\alpha_1^1 \cap \alpha_2^1);(\alpha_1^2 \cap \alpha_2^2) \Rightarrow (\alpha_1^1;\alpha_1^2) \cap (\alpha_2^1;\alpha_2^2)$ and, by the induction hypothesis, there are $\gamma_1^1;\ldots;\gamma_n^1$ and $\gamma_1^2;\ldots;\gamma_m^2$ such that $\gamma_1'^1;\ldots;\gamma_n'^1;\gamma_1'^2;\ldots;\gamma'^2 \Rightarrow (\alpha_1^1 \cap \alpha_2^1);(\alpha_1^2 \cap \alpha_2^2)$ for some instances $\gamma_i'^j$ of γ_i^j. This finishes the proof. □

Lemma 16. *For all points $u \in A$ and all programs α such that $u \xrightarrow{\alpha^\circlearrowleft} u$ and all formulae ψ it holds that: If $[\alpha^\circlearrowleft]\psi \in dom(u)$, then $\psi \in dom(u)$.*

Proof. The proof proceeds similarly to Lemma 15. Again, without loss of generality, the witness graph for $u \xrightarrow{\alpha^\circlearrowleft} u$ is contained in one arena. Instead of converting the program into abstract forward edges, we convert it into abstract loop edges. Once this is done, we invoke Item (3) of Lemma 10. □

Lemma 17 (Existence Lemma)**.** *For any $u \in A$ and any program α, if $\langle\alpha\rangle\varphi \in dom(u)$, then there is a state $v \in A$ with $u \xrightarrow{\alpha} v$, such that $\varphi \in dom(v)$.*

Proof. If $\alpha \in Cyc$, there is an α-loop at u by the construction of $\mathfrak{A}$.

The other case is that $\alpha \in Forw$. Set $X^* = \{\varphi \mid \langle\alpha\rangle\varphi \in dom(u)\}$. This set, in general, is inconsistent. However, the set of nonempty subsets of X that contain φ and all ψ such that $[\alpha]\psi \in \mathrm{dom}(u)$ is nonempty and satisfies the conditions of Zorn's Lemma. Hence, there is a maximal such set Φ. We claim that it is an MCS: For each $\mathrm{PDL}_0^{\cap,?}[\tau]$-formula ψ, either $\langle\alpha\rangle\psi \in \mathrm{dom}(u)$ or $[\alpha]\neg\psi \in \mathrm{dom}(u)$. In the latter case, $\psi \in \Phi$. In the former case, both $\langle\alpha\rangle\psi \in \mathrm{dom}(u)$ and $\langle\alpha\rangle\psi \in \mathrm{dom}(u)$. If neither of these is in Φ, then Φ is not maximal since $\vdash \neg\langle\alpha\rangle\psi \vee \neg\langle\alpha\rangle\psi \leftrightarrow [\alpha]\bot$. So Φ was maximal after all.

Further, let $v = (\Psi, \alpha, u)$. By construction, $\varphi \in dom(v)$ and there is a large program α' such that $u \xrightarrow{\alpha'} v$ is maximally consistent. Thus $u \xrightarrow{\alpha} v$ since there is an α'-edge from u to v. □

Lemma 18. *For all $v \in A$ and for all $\varphi \in \mathrm{PDL}_0^{\cap,?}$ we have $\mathfrak{A}, v \vDash \varphi$ if and only if $\varphi \in dom(v)$.*

Proof. We only prove the if part. The other direction follows from contraposition and the fact that an MCS contains every formula or its negation. The case for atomic propositions is by the definition of the valuations in the canonical model, the case for boolean connectives follows from the closure properties of MCS. The case for $\varphi = \langle\alpha\rangle\psi$ follows from Lemma 17. This leaves the case $\varphi = [\alpha]\psi$, which follows from Lemma 15 and Lemma 16. □

Theorem 19. *$\vdash$ is sound and complete: $\Phi \vdash \varphi$ if and only if $\Phi \vDash \varphi$.*

Proof. Soundness is by Lemma 2. For the sake of contradiction, assume that $\Phi \vDash \varphi$ but $\Phi \nvdash \varphi$. Then $\Phi \cup \{\neg\varphi\}$ is consistent and contained in an MCS Φ'. But the generation 0 node $\mathfrak{A}, \Phi'$ is such that $\mathfrak{A}, \Phi' \vDash \psi$ if and only if $\psi \in \Phi'$. Since $\neg\varphi \in \Phi'$ and $\Phi \subseteq \Phi'$, this contradicts $\Phi \vDash \varphi$. Hence, $\vdash$ is complete. □

6 Conclusion

We have presented a refined construction of a canonical model for the iteration-free fragment of Propositional Dynamic Logic with Intersection and Tests ($\mathrm{PDL}_0^{\cap,?}$) and used this to prove completeness of an axiom system for this logic. The trick that handles the combinatorial difficulties introduced by the interaction between the intersection operator and test programs is the use of several copies of a maximally consistent set for worlds in this Kripke model.

As in turns out, there are parallels between our construction and that in [1], respectively those used for fragments of PDL, e.g. in [2]. Both constructions use multiple copies of maximally consistent sets in their canonical model, and both construct this model as the countable union of partial approximations, each of which is generated by constructing witnesses for all diamond formulae that lack such a witness. The construction in this paper is more explicit and more constructive, for example because it does not rely on a well-ordering of unsatisfied diamond formulae, or the language being countable.

The rules of the proof calculi in this paper and in [1] also have similarities. However, our approach does not rely on computable or recursively enumerable auxiliary functions but rather incorporates their content into the calculus itself.

Future work will attempt to derive a weakly complete axiomatisation for full $\mathrm{PDL}^{\cap,?}$, i.e. the logic including Kleene iteration, based on such a refined canonical model construction over finite sets of formulae.

Acknowledgment. We thank Philippe Balbiani for the discussion we had on the topic of axiomatisations of Iteration-Free PDL with Intersection.

References

[1] P. Balbiani (2003): *Eliminating unorthodox derivation rules in an axiom system for iteration-free PDL with intersection*. *Fundam. Inform.* 56, pp. 211–242.

[2] P. Balbiani & L. Fariñas del Cerro (1998): *Complete axiomatization of a relative modal logic with composition and intersection*. *J. of Applied Non-Classical Logics* 8(4), pp. 325–335, doi:10.1080/11663081.1998.10510949.

[3] P. Balbiani & D. Vakarelov (2001): *Iteration-free PDL with Intersection: a Complete Axiomatization*. *Fundam. Inform.* 45(3), pp. 173–194.

[4] P. Balbiani & D. Vakarelov (2003): *PDL with Intersection of Programs: A Complete Axiomatization*. *J. of Applied Non-Classical Logics* 13(3-4), pp. 231–276, doi:10.3166/jancl.13.231-276.

[5] F. Berman & M. S. Paterson (1981): *Propositional dynamic logic is weaker without tests*. *TCS* 16, pp. 321–328, doi:10.1016/0304-3975(81)90102-X.

[6] S. Danecki (1984): *Nondeterministic Propositional Dynamic Logic with intersection is decidable*. In: *Proc. 5th Symp. on Computation Theory, LNCS* 208, Springer, pp. 34–53, doi:10.1007/3-540-16066-3_5.

[7] M. J. Fischer & R. E. Ladner (1979): *Propositional Dynamic Logic of Regular Programs*. *J. of Comp. and Syst. Sc.* 18(2), pp. 194–211, doi:10.1016/0022-0000(79)90046-1.

[8] D. Kozen & R. Parikh (1981): *An Elementary Proof of the Completeness of PDL*. *TCS* 14, pp. 113 – 118, doi:10.1016/0304-3975(81)90019-0.

[9] M. Lange & C. Lutz (2005): *2-ExpTime Lower Bounds for Propositional Dynamic Logics with Intersection*. *J. of Symbolic Logic* 70(4), pp. 1072–1086, doi:10.2178/jsl/1129642115.

[10] R. Parikh (1978): *The completeness of propositional dynamic logic*. In: *Proc. 7th Symp. on Math. Foundations of Computer Science, FOCS'78, LNCS* 64, Springer, pp. 403–415, doi:10.1007/3-540-08921-7_88.

[11] S. Passy & T. Tinchev (1991): *An essay in combinatory dynamic logic*. *Inform. and Comp.* 93, pp. 263–332, doi:10.1016/0890-5401(91)90026-X.

[12] V. R. Pratt (1976): *Semantical Considerations on Floyd-Hoare Logic*. In: *Proc. 17th Ann. Symp. on Foundations of Computer Science, FOCS'76*, IEEE, pp. 109–121, doi:10.1109/SFCS.1976.27.

[13] V. R. Pratt (1980): *A Near Optimal Method for Reasoning About Action*. *J. of Comp. and Syst. Sc.* 2, pp. 231–254, doi:10.1016/0022-0000(80)90061-6.

[14] H. Prendinger & G. Schurz (1996): *Reasoning about Action and Change. A Dynamic Logic Approach*. *J. of Logic, Language and Information* 5(2), pp. 209–245, doi:10.1007/BF00173701.

[15] R. S. Streett (1982): *Propositional Dynamic Logic of Looping and Converse Is Elementarily Decidable*. *Inform. and Control* 54(1/2), pp. 121–141, doi:10.1016/S0019-9958(82)91258-X.