A Non-wellfounded, Labelled Proof System for Propositional Dynamic Logic

Simon Docherty[1⋆] and Reuben N. S. Rowe[2⋆⋆]

[1] Department of Computer Science, University College London, UK
simon.docherty@ucl.ac.uk
[2] School of Computing, University of Kent, Canterbury, UK
reuben.rowe@kent.ac.uk

Abstract. We define an infinitary labelled sequent calculus for PDL, **G3PDL**$^{\infty}$. A finitarily representable cyclic system, **G3PDL**$^{\omega}$, is then given. We show that both are sound and complete with respect to standard models of PDL and, further, that **G3PDL**$^{\infty}$ is cut-free complete. We additionally investigate proof-search strategies in the cyclic system for the fragment of PDL without tests.

1 Introduction

Fischer and Ladner's Propositional Dynamic Logic (PDL) [14], which is the propositional variant of Pratt's Dynamic Logic [34], is perhaps *the* quintessential modal logic of action. While (P)DL arose initially as a modal logic for reasoning about program execution its impact as a formalism for extending 'static' logical systems with 'dynamics' via composite actions [22, p. 498] has been felt broadly across logic. This is witnessed in extensions and variants designed for reasoning about games [31], natural language [21], cyber-physical systems [33], epistemic agents [19], XML [1], and knowledge representation [11], among others.

Much of the proof theoretic work on PDL, and logics extending it, focuses on Hilbert-style axiomatisations, which are not amenable to automation. Outside of this, proof systems for PDL itself can broadly be characterised as one of two sorts. Falling into the first category are a multitude of infinitary systems [35,24,16] employing either infinitely-wide ω-proof rules, or (equivalently) allowing countably infinite contexts. In the other category are tableau-based algorithms for deciding PDL-satisfiability [18,20]. While these are (neccessarily) finitary, they employ a great deal of auxillary structure tailored to the decision procedure itself.

In the proof theory of modal logic, a high degree of uniformity and modularity has been achieved through labelled systems. The idea of using labels as syntactic representatives of Kripke models in modal logic proof systems can be traced back to Kanger [25], but perhaps has been most famously deployed by Fitting [15]. A succinct history of the use of labelled systems is provided by Negri [29]. Negri's

⋆ supported by EPSRC grant no. EP/S013008/1
⋆⋆ supported by EPSRC grant no. EP/N028759/1.

work [28] is the high point of the technique, giving a procedure to transform frame conditions for Kripke models into labelled sequent calculi rules preserving structural properties of the proof system, given they are defined as *coherent axioms*.

The power of this rule generation technique is of particular interest because it enables the specification of sound and complete systems for classes of Kripke frames that are first-order, but not modally, definable. In the context of PDL-type logics, this is of interest because of common additional program constructs like intersection which have a non-modally definable intended interpretation [32]. However, even with this expressive power, such a framework on its own cannot account for program modalities involving iteration. In short, formulae involving these modalities are interpreted via the reflexive-transitive closure of accessibility relations, and this closure is not first-order (and therefore, not coherently) definable. Something more must be done to capture the PDL family of logics.

In this paper we provide the first step towards a uniform proof theory of the sort that is currently missing for this family of logics by giving two new proof systems for PDL. We combine two ingredients from modern proof theory that have hitherto remained separate: labelled deduction à la Negri and non-wellfounded (in particular, *cyclic*) sequent calculi.

We first construct a labelled sequent calculus $\mathbf{G3PDL}^{\infty}$, extending that of Negri [28], in which proofs are permitted to be infinitely tall. For this system soundness (via descending counter-models) and cut-free completeness (via counter-model construction) are proved in a similar manner to Brotherston and Simpson's infinitary proof theory for first-order logic with inductive definitions [6]. Next we restrict attention to *regular* proofs, meaning only those infinite proof trees that are finitely representable (i.e. only have a finite number of distinct subtrees), obtaining the cyclic system $\mathbf{G3PDL}^{\omega}$. This can be done by permitting the forming of backlinks (or, cycles) in the proof tree, granted a (decidable) trace condition guaranteeing soundness can be established. We then show that the axiomatisation of PDL [23] can be derived in $\mathbf{G3PDL}^{\omega}$, obtaining completeness. We finish the paper with an investigation of proof-search in the cyclic system for a sub-class of sequents, and conjecture cut-free completeness for the test-free fragment of PDL.

There are a number of advantages to setting up PDL's proof theory in this manner. Most crucially, through the cyclic system we obtain a finitary sequent calculus with natural, declarative proof rules, in which requirements on Kripke models and traces are elegantly handled with the labels. Such a system is amenable to automation through, for example, the Cyclist [5] theorem prover. We also conjecture (see Section 5) that labels can be used to compute bounds to determine termination of proof-search.

We believe this work can be built upon in two complementary directions. First, towards a uniform proof theory of PDL-type logics. We conjecture the presence of labels should facilitate the extension of the system with rules for additional program constructs. Second, this work gives a case study for the extension of the expressivity of Negri-style modal proof theory. Our system thus

indicates the viability of constructing an analogous general framework that naturally captures modal logics interpreted on classes of Kripke frames defined by logics more expressive than first-order logic (for example, epistemic logics with a common knowledge modality). We discuss this, and other ideas for future work, in the conclusion.

For space reasons we elide proofs, but these can be found in an extended version of this paper available online [13].

Related Work. Beyond the proof systems outlined above, the most significant related work can be found in Das and Pous' [9,10] cyclic proof systems for deciding Kleene algebra (in)equalities. Das and Pous' insight that iteration can be handled in a cyclic sequent calculus is essential to our work here, although there are additional complications involved in formulating a system for PDL because of the interaction between programs (which form a Kleene algebra with tests) and formulae. We also note that Goré and Widmann's tableau procedure also utilises the formation of cycles in proof trees. Our proof of cut-free completeness of the infinitary system also follows that of Brotherston and Simpson [6] for first-order logic with inductive definitions.

Recent work by Cohen and Rowe [8] gives a cyclic proof system for the extension of first-order logic with a transitive closure operator and we conjecture that our labelled cyclic system (and labelled cyclic systems for modal logics more generally) can be formalised within it. This idea echoes van Benthem's suggestion that the most natural frame language for many modal logics is not first-order logic, but in fact first-order logic with a least fixed point operator [4].

Cyclic proof systems have also been defined for some modal logics with similar model properties to PDL, including the logic of common knowledge [40] and Gödel-Löb logic [37]. The idea of cyclic proof can be traced to modal μ-calculus [30]. Indeed, it can be shown that the logic of common knowledge [2], Gödel-Löb logic [4,39] and PDL [4,7] can be faithfully interpreted in the modal μ-calculus, indicating that perhaps cyclic proof was the right approach for PDL all along.

2 PDL: Syntax and Semantics

The syntax of PDL formulas is defined as follows. We assume countably many atomic *propositions* (ranged over by p, q, r), and countably many atomic *programs* (ranged over by a, b, c).

Definition 1 (Syntax of PDL). *The set of* formulas *(φ, ψ, ...) and the set of* programs *(α, β, ...) are defined mutually by the following grammar:*

$$\varphi, \psi ::= \bot \mid p \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \varphi \to \psi \mid [\alpha]\varphi$$
$$\alpha, \beta, \gamma ::= a \mid \alpha \,;\, \beta \mid \alpha \cup \beta \mid \varphi? \mid \alpha^*$$

We briefly reprise the semantics of PDL (see [23, §5.2]). A PDL model $\mathfrak{m} = (\mathcal{S}, \mathcal{I})$ is a Kripke model consisting of a set $\mathcal{S}$ of *states* and an *interpretation* function $\mathcal{I}$ that assigns: a subset of $\mathcal{S}$ to each atomic proposition; and

a binary relation on $\mathcal{S}$ to each atomic program. We inductively construct an extension of the interpretation function, denoted $\mathcal{I}_{\mathfrak{m}}$, that operates on the full set of propositions and programs.

Definition 2 (Semantics of PDL). *Let $\mathfrak{m} = (\mathcal{S}, \mathcal{I})$ be a PDL model. We define the extended interpretation function $\mathcal{I}_{\mathfrak{m}}$ inductively as follows:*

$$\begin{aligned}
\mathcal{I}_{\mathfrak{m}}(\bot) &= \emptyset & \mathcal{I}_{\mathfrak{m}}(a) &= \mathcal{I}(a) \\
\mathcal{I}_{\mathfrak{m}}(p) &= \mathcal{I}(p) & \mathcal{I}_{\mathfrak{m}}(\alpha\,;\beta) &= \mathcal{I}_{\mathfrak{m}}(\alpha) \circ \mathcal{I}_{\mathfrak{m}}(\beta) \\
\mathcal{I}_{\mathfrak{m}}(\varphi \wedge \psi) &= \mathcal{I}_{\mathfrak{m}}(\varphi) \cap \mathcal{I}_{\mathfrak{m}}(\psi) & \mathcal{I}_{\mathfrak{m}}(\alpha \cup \beta) &= \mathcal{I}_{\mathfrak{m}}(\alpha) \cup \mathcal{I}_{\mathfrak{m}}(\beta) \\
\mathcal{I}_{\mathfrak{m}}(\varphi \vee \psi) &= \mathcal{I}_{\mathfrak{m}}(\varphi) \cup \mathcal{I}_{\mathfrak{m}}(\psi) & \mathcal{I}_{\mathfrak{m}}(\varphi?) &= \mathsf{Id}(\mathcal{I}_{\mathfrak{m}}(\varphi)) \\
\mathcal{I}_{\mathfrak{m}}(\varphi \to \psi) &= (\mathcal{S} \setminus \mathcal{I}_{\mathfrak{m}}(\varphi)) \cup \mathcal{I}_{\mathfrak{m}}(\psi) & \mathcal{I}_{\mathfrak{m}}(\alpha^*) &= \bigcup_{k \geq 0} \mathcal{I}_{\mathfrak{m}}(\alpha)^k \\
\mathcal{I}_{\mathfrak{m}}([\alpha]\varphi) &= \mathcal{S} \setminus \mathit{\Pi}_1(\mathcal{I}_{\mathfrak{m}}(\alpha) \circ \mathsf{Id}(\mathcal{S} \setminus \mathcal{I}_{\mathfrak{m}}(\varphi)))
\end{aligned}$$

where $\circ$ denotes relational composition, R^n denotes the composition of R with itself n times, $\mathit{\Pi}_1$ returns a set by projecting the first component of each tuple in a relation, and $\mathsf{Id}(X)$ denotes the identity relation over the set X.

We write $\mathfrak{m}, s \models \varphi$ to mean $s \in \mathcal{I}_{\mathfrak{m}}(\varphi)$, and $\mathfrak{m} \models \varphi$ to mean that $\mathfrak{m}, s \models \varphi$ for all states $s \in \mathcal{S}$. A PDL formula φ is *valid* when $\mathfrak{m} \models \varphi$ for all models $\mathfrak{m}$.

3 An Infinitary, Labelled Sequent Calculus

We now define a sequent calculus for deriving theorems (i.e. valid formulas) of PDL. This proof system has two important features. The first is that it is a *labelled* proof system. Thus sequents contain assertions about the structure of the underlying Kripke models and formulas are labelled with atoms denoting specific states in which they should be interpreted. Secondly, we allow proofs of infinite height.

We assume a countable set $\mathcal{L}$ of *labels* (ranged over by x, y, z) that we will use to denote particular states. A *relational atom* is an expression of the form $x\ R_a\ y$, where x and y are labels and a is an atomic program. A *labelled formula* is an expression of the form $x : \varphi$, where x is a label and φ is a formula. We define a label substitution operation by $z\{x/y\} = y$ when $x = z$, and $z\{x/y\} = z$ otherwise. We lift this to relational atoms and labelled formulas by: $(z\ R_a\ z')\{x/y\} = z\{x/y\}\ R_a\ z'\{x/y\}$ and $(z : \varphi)\{x/y\} = z\{x/y\} : \varphi$.

Sequents are expressions of the form $\mathit{\Gamma} \Rightarrow \mathit{\Delta}$, where $\mathit{\Gamma}$ and $\mathit{\Delta}$ are finite sets of relational atoms and labelled formulas. We denote an arbitrary member of such a set using A, B, etc. As usual, $\mathit{\Gamma}, A$ and $A, \mathit{\Gamma}$ both denote the set $\{A\} \cup \mathit{\Gamma}$, and $\mathit{\Gamma}\{z/y\}$ denotes the application of the (label) substitution $\{x/y\}$ to all the elements in $\mathit{\Gamma}$. We denote by $[\alpha]\mathit{\Gamma}$ the set of formulas obtained from $\mathit{\Gamma}$ by prepending the modality $[\alpha]$ to every labelled formula. That is, we define $[\alpha]\mathit{\Gamma} = \{x\ R_a\ y \mid x\ R_a\ y \in \mathit{\Gamma}\} \cup \{x : [\alpha]\varphi \mid x : \varphi \in \mathit{\Gamma}\}$. $\mathsf{labs}(\mathit{\Gamma})$ denotes the set of all labels ocurring in the relational atoms and labelled formulas in $\mathit{\Gamma}$.

$$\text{(Ax)}\ \frac{}{A \Rightarrow A} \qquad \text{(}\bot\text{)}\ \frac{}{x : \bot \Rightarrow} \qquad \text{(WL)}\ \frac{\Gamma \Rightarrow \Delta}{A, \Gamma \Rightarrow \Delta} \qquad \text{(WR)}\ \frac{\Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, A}$$

$$\text{(}\wedge\text{L)}\ \frac{x : \varphi, x : \psi, \Gamma \Rightarrow \Delta}{x : \varphi \wedge \psi, \Gamma \Rightarrow \Delta} \qquad \text{(}\wedge\text{R)}\ \frac{\Gamma \Rightarrow \Delta, x : \varphi \quad \Gamma \Rightarrow \Delta, x : \psi}{\Gamma \Rightarrow \Delta, x : \varphi \wedge \psi}$$

$$\text{(}\vee\text{L)}\ \frac{x : \varphi, \Gamma \Rightarrow \Delta \quad x : \psi, \Gamma \Rightarrow \Delta}{x : \varphi \vee \psi, \Gamma \Rightarrow \Delta} \qquad \text{(}\vee\text{R)}\ \frac{\Gamma \Rightarrow \Delta, x : \varphi, x : \psi}{\Gamma \Rightarrow \Delta, x : \varphi \vee \psi}$$

$$\text{(}\to\text{L)}\ \frac{\Gamma \Rightarrow \Delta, x : \varphi \quad x : \psi, \Gamma \Rightarrow \Delta}{x : \varphi \to \psi, \Gamma \Rightarrow \Delta} \qquad \text{(}\to\text{R)}\ \frac{x : \varphi, \Gamma \Rightarrow \Delta, x : \psi}{\Gamma \Rightarrow \Delta, x : \varphi \to \psi}$$

$$\text{(}\Box\text{L)}\ \frac{y : \varphi, \Gamma \Rightarrow \Delta}{x : [a]\varphi, x\ R_a\ y, \Gamma \Rightarrow \Delta} \qquad \text{(}\Box\text{R)}\ \frac{x\ R_a\ y, \Gamma \Rightarrow \Delta, y : \varphi}{\Gamma \Rightarrow \Delta, x : [a]\varphi}\ (y \text{ fresh})$$

$$\text{(;L)}\ \frac{x : [\alpha][\beta]\varphi, \Gamma \Rightarrow \Delta}{x : [\alpha\,;\beta]\varphi, \Gamma \Rightarrow \Delta} \qquad \text{(;R)}\ \frac{\Gamma \Rightarrow \Delta, x : [\alpha][\beta]\varphi}{\Gamma \Rightarrow \Delta, x : [\alpha\,;\beta]\varphi}$$

$$\text{(}\cup\text{L)}\ \frac{x : [\alpha]\varphi, x : [\beta]\varphi, \Gamma \Rightarrow \Delta}{x : [\alpha \cup \beta]\varphi, \Gamma \Rightarrow \Delta} \qquad \text{(}\cup\text{R)}\ \frac{\Gamma \Rightarrow \Delta, x : [\alpha]\varphi \quad \Gamma \Rightarrow \Delta, x : [\beta]\varphi}{\Gamma \Rightarrow \Delta, x : [\alpha \cup \beta]\varphi}$$

$$\text{(?L)}\ \frac{\Gamma \Rightarrow \Delta, x : \varphi \quad x : \psi, \Gamma \Rightarrow \Delta}{x : [\varphi?]\psi, \Gamma \Rightarrow \Delta} \qquad \text{(?R)}\ \frac{x : \varphi, \Gamma \Rightarrow \Delta, x : \psi}{\Gamma \Rightarrow \Delta, x : [\varphi?]\psi}$$

$$\text{(}*\text{L)}\ \frac{x : \varphi, x : [\alpha][\alpha^*]\varphi, \Gamma \Rightarrow \Delta}{x : [\alpha^*]\varphi, \Gamma \Rightarrow \Delta} \qquad \text{(}*\text{R)}\ \frac{\Gamma \Rightarrow \Delta, x : \varphi \quad \Gamma \Rightarrow \Delta, x : [\alpha][\alpha^*]\varphi}{\Gamma \Rightarrow \Delta, x : [\alpha^*]\varphi}$$

$$\text{(Subst)}\ \frac{\Gamma \Rightarrow \Delta}{\Gamma\{x/y\} \Rightarrow \Delta\{x/y\}} \qquad \text{(Cut)}\ \frac{\Gamma \Rightarrow \Delta, A \quad A, \Sigma \Rightarrow \Pi}{\Gamma, \Sigma \Rightarrow \Delta, \Pi}$$

Fig. 1: Inference rules of **G3PDL**$^\infty$

We interpret sequents with respect to PDL models using label *valuations* v, which are functions from labels to states. We write $\mathfrak{m}, v \models x\ R_a\ y$ to mean that $(v(x), v(y)) \in \mathcal{I}_\mathfrak{m}(a)$. We write $\mathfrak{m}, v \models x : \varphi$ to mean $\mathfrak{m}, v(x) \models \varphi$. For a sequent $\Gamma \Rightarrow \Delta$, denoted by S, we write $\mathfrak{m}, v \models S$ to mean that $\mathfrak{m}, v \models B$ for *some* $B \in \Delta$ whenever $\mathfrak{m}, v \models A$ for *all* $A \in \Gamma$. We write $\mathfrak{m}, v \not\models S$ whenever this is not the case, i.e. when $\mathfrak{m}, v \models A$ for all $A \in \Gamma$ and $\mathfrak{m}, v \not\models B$ for all $B \in \Delta$. We say S is *valid*, and write $\models S$, when $\mathfrak{m}, v \models S$ for all models $\mathfrak{m}$ and valuations v that map each label to some state of $\mathfrak{m}$.

The sequent calculus **G3PDL**$^\infty$ is defined by the inference rules in fig. 1. A *pre-proof* is a possibly infinite derivation tree built from these inference rules.

Definition 3 (Pre-proof). *A pre-proof is a possibly infinite (i.e. non-well-founded) derivation tree formed from inference rules. A path in a pre-proof is*

a possibly infinite sequence of sequents $s_0, s_1, \ldots (, s_n)$ such that s_0 is the root sequent of the proof, and s_{i+1} is a premise of s_i for each $i < n$.

Not all pre-proofs derive sound judgements.

Example 1. The following pre-proof derives an invalid sequent.

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \Rightarrow x : [a^*]p\end{array}}{\Rightarrow x : [a^*]p, x : p}\,(\mathrm{WR}) \qquad \dfrac{\begin{array}{c}\vdots\\ \Rightarrow x : [a^*]p\end{array}}{\Rightarrow x : [a^*]p, x : [a][a^*]p}\,(\mathrm{WR})}{\Rightarrow x : [a^*]p}\,(*\mathrm{R})$$

Note that, since our sequents consist of *sets* of formulas, each instance of the ($*$R) rule incorporates a contraction

To distinguish pre-proofs deriving valid sequents, we define the notion of a trace through a pre-proof. Traces consist of trace values, which (uniquely) identify particular modalities within labelled formulas. $\boldsymbol{\alpha}_n$ denotes a sequence $\alpha_1, \ldots, \alpha_n$, and ε denotes the empty sequence. We sometimes omit the subscript indicating length, writing $\boldsymbol{\alpha}$, when irrelevant or evident from the context.

Definition 4 (Trace Value). *A* trace value *τ is a tuple $(x, \boldsymbol{\alpha}_n, \beta, \varphi)$ consisting of a label x, a (possibly empty) sequence $\boldsymbol{\alpha}_n$ of n programs, a program β, and a formula φ. We call $\boldsymbol{\alpha}$ the* spine *of τ, and β the* focus *of τ. We write $[\gamma]\tau$ for the trace value $(x, \gamma \cdot \boldsymbol{\alpha}_n, \beta, \varphi)$, and $y : \tau$ for the trace value $(y, \boldsymbol{\alpha}_n, \beta, \varphi)$. In an abuse of notation we also use τ to denote the corresponding labelled formula $x : [\alpha_1] \ldots [\alpha_n][\beta^*]\varphi$.*

Trace values in the conclusion of an inference rule are related to trace values in its premises as follows.

Definition 5 (Trace Pairs). *Let τ and τ' be trace values, with sequents $\Gamma \Rightarrow \Delta$ and $\Gamma' \Rightarrow \Delta'$ (respectively denoted by s and s') the conclusion and a premise, respectively, of an inference rule r; we say that (τ, τ') is a trace pair for (s, s') when $\tau \in \Delta$ and $\tau' \in \Delta'$ and the following conditions hold.*

(1) If τ is the principal formula of the rule instance, then τ' is its immediate ancestor and moreover if the rule is an instance of:
- *($\Box$R) then $\tau = x : [a]\tau'$, where x is the label of the principal formula;*
- *(?R) then $\tau = [\varphi?]\tau'$;*
- *(;R) then $\tau = [\alpha\,;\beta]\tau''$ and $\tau' = [\alpha][\beta]\tau''$ for some trace value τ'';*
- *($\cup$R) then there is some τ'' such that: $\tau = [\alpha \cup \beta]\tau''$; $\tau' = [\alpha]\tau''$ if s' is the left-hand premise; and $\tau' = [\beta]\tau''$ if s' is the right-hand premise;*
- *($*$R) then $\tau = [\alpha^*]\tau'$ if s' is the left-hand premise, and $\tau' = [\alpha]\tau$ if s' is the right-hand premise.*

(2) If τ is not the principal formula of the rule then $\tau = x : \tau'$ if the rule is an instance of (Subst) and x is the label substituted, and $\tau = \tau'$ otherwise.

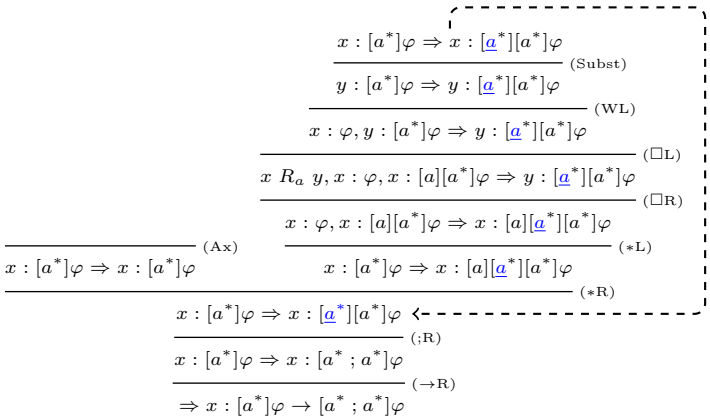


Fig. 2: Representation of a **G3PDL**$^\infty$ proof of $[a^*]\varphi \to [a^*\,;a^*]\varphi$.

If τ is the principal formula of the rule instance and the spine of τ is empty, then we say that the trace pair is progressing.

Notice that when a trace pair is progressing for (s, s'), it is necessarily the case that the corresponding rule is an instance of ($*$R) and that s' is the right-hand premise (although, not necessarily vice versa).

Traces along paths in a pre-proof consist of consecutive pairs of trace values for each corresponding step of the path.

Definition 6 (Trace). *A* trace *is a (possibly infinite) sequence of trace values. We say that a trace $\tau_1, \tau_2, \ldots (, \tau_n)$* follows *a path $s_1, s_2, \ldots (, s_m)$ in a pre-proof when there exists some $k \geq 0$ such that each consecutive pair of trace values (τ_i, τ_{i+1}) is a trace pair for (s_{i+k}, s_{i+k+1}); when $k = 0$, we say that the trace* covers *the path. We say that the trace* progresses *at i if (τ_i, τ_{i+1}) is progressing, and say the trace is infinitely progressing if it progresses at infinitely many points.*

Proofs are pre-proofs that satisfy a well-formedness condition, called the global trace condition.

Definition 7 (Infinite Proof). *A* **G3PDL**$^\infty$ proof *is a pre-proof in which every infinite path is followed by some infinitely progressing trace.*

Example 2. Figure 2 shows a finite representation of a **G3PDL**$^\infty$ proof of the formula $[a^*]\varphi \to [a^*\,;a^*]\varphi$. The full infinite proof can be obtain by unfolding the cycle an infinite number of times. An infinitely progressing trace following the (unique) infinite path in this proof is indicated by the underlined programs highlighted in blue, which denote the focus of the trace value in each sequent. The progression point is the (only) instance of the ($*$R) rule.

Figure 3 shows a finite representation of a **G3PDL**$^\infty$ proof of the sequent $x : [a^*]\varphi \Rightarrow x : [(a^*)^*]\varphi$. This proof is more complex than that of fig. 2, and involves two overlapping cycles. This proof contains more than one infinite path (in

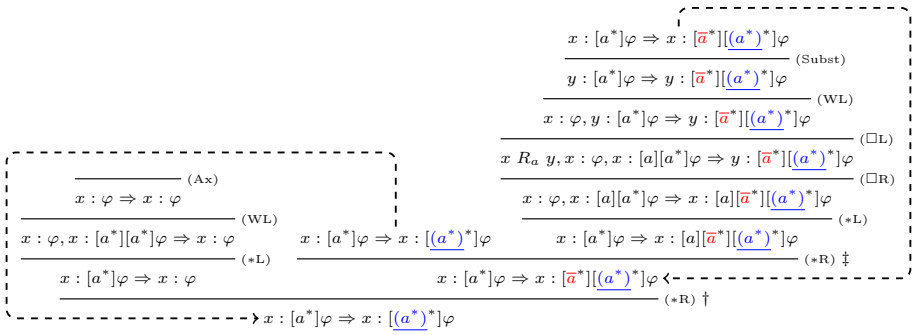


Fig. 3: Representation of a **G3PDL**$^\infty$ proof of $x : [a^*]\varphi \Rightarrow x : [(a^*)^*]\varphi$.

fact, it contains an infinite number of infinite paths). However, they fall into three categories: (1) those that eventually traverse only the upper cycle; (2) those that eventually traverse only the lower cycle; and (3) those that traverse both cycles infinitely often. Infinite paths of the first variety have an infinitely progressing trace indicated by the overlined programs highlighted in red. The progression point is the upper instance of (∗R) rule, marked by (‡). The remaining infinite paths have a trace indicated by the underlined programs highlighted in blue. This trace does not progress around the upper cycle (for those paths that traverse it), but does progress once around each lower cycle at the instance of the (∗R) rule marked by (†). Since these paths traverse this lower cycle infinitely often, the trace is infinitely progressing.

Remark 1. The notion of trace in the system for Kleene Algebra of Das and Pous [9,10] appears simpler than ours: a sequence of formulas (on the left) connected by ancestry, with such a trace being valid if it is principal for a (left) unfolding rule infinitely often. In fact, we can show that our definition of trace is equivalent to an analogous formulation of this notion for our system. However, our definition allows for a direct, semantic proof of soundness via infinite descent. In contrast, the soundness proof in [10] relies on cut-admissibility and an inductive proof-theoretic argument for the soundness of the cut-free fragment. It is unclear that a similar technique can be used to show soundness of the cut-free fragment of our system. Furthermore, the cut-free fragment of the system of Das and Pous is notable in that it admits a simpler trace condition than the full system: namely, that every infinite path is fair for the (left) unfolding rule [10, prop. 8]. Our system does not satisfy this property, due to the ability to perform contraction and weakening, as demonstrated in example 1.

The proof system is sound since, for invalid sequents, we can map traces to decreasing sets of counter-examples in (finitely branching) models.

A path in a model $\mathfrak{m}$ is a sequence of states $s_1, \ldots, s_n$ in $\mathfrak{m}$ such that each successive pair of states satisfies $(s_i, s_{i+1}) \in \mathcal{I}_\mathfrak{m}(a)$ for some a. A path in $\mathfrak{m}$ is called *loop-free* if it does not contain any repeated states. If $\boldsymbol{s}$ and $\boldsymbol{s'}$ are paths in $\mathfrak{m}$, we write $\boldsymbol{s} \sqsubseteq \boldsymbol{s'}$ to denote that $\boldsymbol{s}$ is a prefix of $\boldsymbol{s'}$.

An m-partition of a path $\boldsymbol{s}_n$ is a sequence of m increasing indices $k_1 \leq \ldots \leq k_m \leq n$. A path in $\mathfrak{m}$ for a trace value $\tau = (x, \boldsymbol{\alpha}_n, \beta, \varphi)$ with respect to a valuation v is a path $\boldsymbol{s}_m$ in $\mathfrak{m}$ with $s_1 = v(x)$ having an n-partition $k_1, \ldots, k_n$ satisfying $(s_{k_i}, s_{k+1}) \in \mathcal{I}_{\mathfrak{m}}(\alpha_{i+1})$ for each $0 \leq i < n$ and $(s_{k_n}, s_m) \in \mathcal{I}_{\mathfrak{m}}(\beta^*)$, where we take $k_0 = 1$ (i.e. $s_{k_0} = s_1$). The n-partition $k_1, \ldots, k_n$ is called a partition of $\boldsymbol{s}_m$ for τ. A counter-example in $\mathfrak{m}$ for a trace value τ at v is simply a path $\boldsymbol{s}_m$ in $\mathfrak{m}$ for τ w.r.t. v such that $\mathfrak{m}, s_m \not\models \varphi$.

A given path in $\mathfrak{m}$ for τ at v can, in general, have many different partitions. A partition $\boldsymbol{k}_n$ of a path $\boldsymbol{s}_m$ for τ at v is called *maximal* if the length of its final segment $s_{k_n}, \ldots, s_m$ is maximal among all such partitions. We define the *weight* of a path $\boldsymbol{s}$ in $\mathfrak{m}$ for τ at v to be the length of the final segment(s) of its maximal partition(s). We denote this by $\mu_{(\mathfrak{m},v)}(\boldsymbol{s}, \tau)$. If Π is a set of paths in $\mathfrak{m}$ for τ at v, we define the *measure* of Π, denoted $\mu_{(\mathfrak{m},v)}(\Pi, \tau)$, to be the multiset of weights of the paths it contains; that is $\mu_{(\mathfrak{m},v)}(\Pi, \tau) = \{\mu_{(\mathfrak{m},v)}(\boldsymbol{s}, \tau) \mid \boldsymbol{s} \in \Pi\}$.

The measure for trace values in a model $\mathfrak{m}$ at a valuation v, then, is simply the measure of the set of all of its 'nearest' counter-examples.

Definition 8 (Trace Value Measure). *Let $\mathcal{C}_{(\mathfrak{m},v)}(\tau)$ denote the set of all loop-free counter-examples $\boldsymbol{s}$ in $\mathfrak{m}$ for τ at v such that there is no counter-example $\boldsymbol{s'}$ in $\mathfrak{m}$ for τ at v with $\boldsymbol{s'} \sqsubseteq \boldsymbol{s}$. The measure of τ in $\mathfrak{m}$ at v is defined as $\mu_{(\mathfrak{m},v)}(\tau) = \mu_{(\mathfrak{m},v)}(\mathcal{C}_{(\mathfrak{m},v)}(\tau), \tau)$.*

For finitely branching models $\mathfrak{m}$, it is clear that trace value measures are always finite. Note that finite multisets M of elements of a well-ordering can be well-ordered using, e.g., the Dershowitz-Manna ordering $<_{\mathrm{DM}}$ [12]. This means that we have the following property.

Lemma 1 (Descending Counter-models). *Let $\Gamma \Rightarrow \Delta$, denoted S, be the conclusion of an instance of an inference rule, and suppose there is a finitely branching model $\mathfrak{m}$ and valuation v such that $\mathfrak{m}, v \not\models S$, then there is a premise $\Gamma' \Rightarrow \Delta'$ of the rule instance, denoted S', and a valuation v' such that $\mathfrak{m}, v' \not\models S'$ and for each trace pair (τ, τ') for (S, S'), $\mu_{(\mathfrak{m},v')}(\tau') \leq_{DM} \mu_{(\mathfrak{m},v)}(\tau)$ and also $\mu_{(\mathfrak{m},v')}(\tau') <_{DM} \mu_{(\mathfrak{m},v)}(\tau)$ if (τ, τ') is progressing.*

This entails the soundness of our proof system, since PDL has the finite model property [14, Thm. 3.2]. This property states that, if a PDL formula is satisfiable, then it is satisfiable in a finite (and thus finitely branching) model. Thus, if a sequent is not valid then there is a finitely branching model that falsifies it. If a **G3PDL$^\infty$** proof $\mathcal{P}$ were to derive an invalid sequent, then by lemma 1 it would contain an infinite path $\Gamma_1 \Rightarrow \Delta_1, \Gamma_2 \Rightarrow \Delta_2, \ldots$ for which there exists a finite model $\mathfrak{m}$ and a matching sequence of valuations $v_1, v_2, \ldots$ that invalidate each sequent in the path. Moreover, these invalidating valuations ensure that the measures of the trace values in any trace pair along the path is decreasing, and strictly so for progressing trace pairs. However, since $\mathcal{P}$ is a proof, it satisfies the global trace condition. This means that there would be an infinitely progressing trace following the path $\Gamma_1 \Rightarrow \Delta_1, \Gamma_2 \Rightarrow \Delta_2, \ldots$ and thus we would be able to construct an infinitely descending chain of (finite) trace

value measures. Because the set of finite trace value measures is well-founded, this is impossible and so the derived sequent must in fact be valid.

Theorem 1 (Soundness). $\mathbf{G3PDL}^{\infty}$ *derives only valid sequents.*

The *cyclic* system $\mathbf{G3PDL}^{\omega}$ is obtained by restricting consideration to only those proofs of $\mathbf{G3PDL}^{\infty}$ that are *regular*, i.e. have only a finite number of distinct subtrees.

Definition 9 (Cyclic Pre-proof). *A* cyclic *pre-proof is a pair* (P, f) *consisting of a finite derivation tree* P *possibly containing open leaves called* buds*, and a function* f *assigning to each bud an internal node of the tree, called its* companion*, with a syntactically identical sequent.*

We usually represent a cyclic pre-proof as the graph induced by identifying each bud with its companion (as in figs. 2 and 3). The infinite unfolding of a cyclic pre-proof is the $\mathbf{G3PDL}^{\infty}$ pre-proof obtained as the limit of the operation that replaces each bud with a copy of the subderivation concluding with its companion an infinite number of times. A cyclic proof is a cyclic pre-proof whose infinite unfolding satisfies the global trace condition. As in other cyclic systems (e.g. [6,8,36,38]) it is decidable whether or not this is the case via a construction involving complementation of Büchi automata. This means that decidability of the global trace condition for $\mathbf{G3PDL}^{\omega}$ pre-proofs is PSPACE-complete.

Since every $\mathbf{G3PDL}^{\omega}$ is also a $\mathbf{G3PDL}^{\infty}$ proof, soundness of the cyclic system is an immediate corollary of theorem 1.

Corollary 1. *If* $\Gamma \Rightarrow \Delta$ *is derivable in* $\mathbf{G3PDL}^{\omega}$*, then* $\Gamma \Rightarrow \Delta$ *is valid.*

4 Completeness

In this section, we give completeness results for our systems. We show that the full system, $\mathbf{G3PDL}^{\infty}$, is cut-free complete. On the other hand, if we allow instances of the (Cut) rule, then every valid theorem of PDL has a proof in the cyclic subsystem $\mathbf{G3PDL}^{\omega}$.

4.1 Cut-free Completeness of $\mathbf{G3PDL}^{\infty}$

We use a standard technique of defining a pre-proof that encodes an exhaustive search for a cut-free proof (as used in, e.g., [6,8]). For invalid sequents, this results in a pre-proof from which we can construct a counter-model, using the formulas that occur along a particular path.

A *schedule* σ is an enumeration of labelled non-atomic formulas in which each labelled formula occurs infinitely often. The i^{th} element of σ is written σ_i.

Definition 10 (Search Tree). *Given a sequent* $\Gamma \Rightarrow \Delta$ *and a schedule* σ*, we can define an infinite sequence* $\boldsymbol{\mathcal{D}}$ *of open derivations inductively. Taking* $\mathcal{D}_0 = \Gamma \Rightarrow \Delta$*, we construct each* $\mathcal{D}_{i+1}$ *from its predecessor* $\mathcal{D}_i$ *by:*

1. firstly closing any open leaves $\Gamma' \Rightarrow \Delta'$ for which $x : \bot \in \Gamma$ for some x or $\Gamma \cap \Delta \neq \emptyset$ by applying weakening rules leading to an instance of $(\bot)$ or an axiom $A \Rightarrow A$ for some $A \in \Gamma \cap \Delta$ (thus the antecedent of each remaining open node is disjoint from its consequent);

2. then replacing each remaining open node $\Gamma' \Rightarrow \Delta'$ in which σ_i occurs with applications of the rule for which σ_i is principal in the following way.

- *If $\sigma_i = x : [a]\varphi \in \Delta'$, then we pick a label y not ocurring in $\Gamma' \Rightarrow \Delta'$, and replace the open node with the following derivation.*

$$\frac{x\ R_a\ y, \Gamma' \Rightarrow \Delta', y : \varphi}{\Gamma' \Rightarrow \Delta', x : [a]\varphi}\ (\Box\mathrm{R})$$

- *If $\sigma_i = x : [a]\varphi \in \Gamma'$ then, letting $\{y_1, \ldots, y_n\}$ be the set of all y_i such that $x\ R_a\ y_i \in \Gamma'$, we replace the open node with the following derivation.*

$$\dfrac{\begin{array}{c} x : [a]\varphi, \{y_1 : \varphi, \ldots, y_n : \varphi\}, \Gamma' \Rightarrow \Delta' \\ \vdots \\ x : [a]\varphi, \{y_1 : \varphi, y_2 : \varphi\}, \Gamma' \Rightarrow \Delta' \end{array}}{x : [a]\varphi, \{y_1 : \varphi\}, \Gamma' \Rightarrow \Delta'}\ (\Box\mathrm{L})}{x : [a]\varphi, \Gamma' \Rightarrow \Delta'}\ (\Box\mathrm{L})$$

- *In all other cases, we replace the open node with an application of the appropriate rule (r) as follows, where Γ'_i and Δ'_i, $i \in \{1, 2\}$, are the sets of left and right immediate ancestors of σ_i, respectively, for the appropriate premise.*

$$\frac{\Gamma'_1, \Gamma' \Rightarrow \Delta', \Delta'_1 \quad (\Gamma'_2, \Gamma' \Rightarrow \Delta', \Delta'_2)}{\Gamma' \Rightarrow \Delta'}\ (r)$$

Since each $\mathcal{D}_i$ is a prefix of $\mathcal{D}_{i+1}$, there is a smallest derivation containing each $\mathcal{D}_i$ as a prefix. We call this derivation a search tree *for $\Gamma \Rightarrow \Delta$ (w.r.t. σ).*

Notice that search trees do not contain instances of the (Cut) or (Subst) rules. Moreover, when a given search tree $\mathcal{D}$ is not a valid proof, we may extract from it two sets of labelled formulas and relational atoms that we can use to construct a countermodel. If $\mathcal{D}$ is not a valid proof, then either it contains an open node to which no schedule element applies or it contains an infinite path that does not satisfy the global trace condition (an *untraceable* branch). For a search tree $\mathcal{D}$, we say that a pair (Γ, Δ) is a *template induced by* $\mathcal{D}$ when either: (i) $\Gamma \Rightarrow \Delta$ is an open node of $\mathcal{D}$; or (ii) $\Gamma = \bigcup_{i>0} \Gamma_i$ and $\Delta = \bigcup_{i>0} \Delta_i$, where $\Gamma_1 \Rightarrow \Delta_1, \Gamma_2 \Rightarrow \Delta_2, \ldots$ is an untraceable branch in $\mathcal{D}$. Notice that, due to the construction of search trees, the component sets of a template are necessarily disjoint. Given a template, we construct a PDL model as follows.

Definition 11 (Countermodel Construction). *Let $P = (\Gamma, \Delta)$ be a template induced by a search tree. The PDL model determined by the template P is given by $\mathfrak{m}_P = (\mathcal{L}, \mathcal{I}_P)$, where $\mathcal{I}_P$ is the following interpretation function:*

$$[\alpha](\varphi \to \psi) \to ([\alpha]\varphi \to [\alpha]\psi) \quad (1) \qquad [\alpha](\varphi \wedge \psi) \to ([\alpha]\varphi \wedge [\alpha]\psi) \quad (2)$$

$$[\alpha \cup \beta]\varphi \leftrightarrow [\alpha]\varphi \wedge [\beta]\varphi \quad (3) \qquad [\alpha\,;\beta]\varphi \leftrightarrow [\alpha][\beta]\varphi \quad (4)$$

$$[\psi?]\varphi \leftrightarrow (\psi \to \varphi) \quad (5) \qquad \varphi \wedge [\alpha][\alpha^*]\varphi \leftrightarrow [\alpha^*]\varphi \quad (6)$$

$$\varphi \wedge [\alpha^*](\varphi \to [\alpha]\varphi) \to [\alpha^*]\varphi \quad (7)$$

$$\frac{\varphi \quad \varphi \to \psi}{\psi}\ (\mathsf{MP}) \qquad \frac{\varphi}{[\alpha]\varphi}\ (\mathsf{Nec})$$

Fig. 4: Axiomatisation of PDL.

1. $\mathcal{I}_P(p) = \{x \mid x : p \in \Gamma\}$ *for each atomic proposition* p*; and*
2. $\mathcal{I}_P(a) = \{(x, y) \mid x\ R_a\ y \in \Gamma\}$ *for each atomic program* a*.*

We write $\mathfrak{v}$ *for the valuation defined by* $\mathfrak{v}(x) = x$ *for each label* x*.*

PDL models determined by templates have the following property.

Lemma 2. *Let* $P = (\Gamma, \Delta)$ *be a template induced by a search tree. Then we have* $\mathfrak{m}_P, \mathfrak{v} \models A$ *for all* $A \in \Gamma$ *and* $\mathfrak{m}_P, \mathfrak{v} \not\models B$ *for all* $B \in \Delta$*.*

Lemma 2 entails the cut-free completeness of $\mathbf{G3PDL}^{\infty}$.

Theorem 2 (Completeness of $\mathbf{G3PDL}^{\infty}$). *If* $\Gamma \Rightarrow \Delta$ *is valid, then it has a cut-free* $\mathbf{G3PDL}^{\infty}$ *proof.*

4.2 Completeness of $\mathbf{G3PDL}^{\omega}$ for PDL

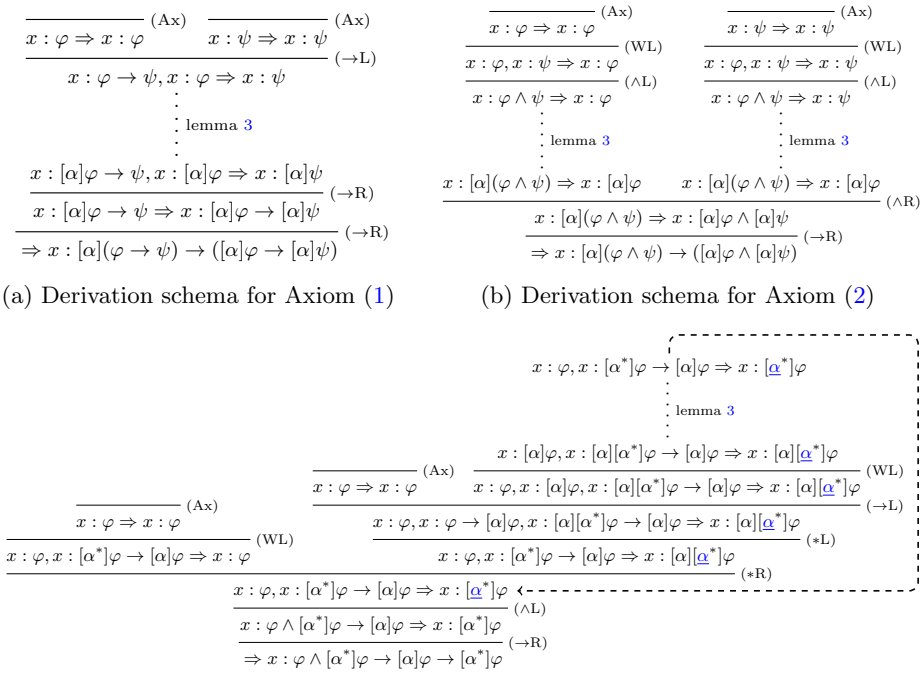
We show that the cyclic system $\mathbf{G3PDL}^{\omega}$ can derive all theorems of PDL by demonstrating that it can derive each of the axiom schemas and inference rules in fig. 4, which (along with the axiom schemas of classical propositional logic) constitute a complete axiomatisation of PDL [23, §7.1].

The derivation of the axioms of classical propositional logic is standard, and axioms (3) to (6) are immediately derivable via the left and right proof rules for their corresponding syntactic constructors. Each such derivation is finite, and thus trivially a $\mathbf{G3PDL}^{\omega}$ proof. Axioms (1), (2), (7) and (Nec) require the following lemma showing that a general form of necessitation is derivable.

Lemma 3 (Necessitation). *For any labelled formula* $x : \varphi$*, program* α*, and finite set* Γ *of labelled formulas such that* $\mathsf{labs}(\Gamma) = \{x\}$*, there exists a* $\mathbf{G3PDL}^{\omega}$ *derivation concluding with the sequent* $[\alpha]\Gamma \Rightarrow x : [\alpha]\varphi$ *and containing open leaves of the form* $\Gamma \Rightarrow x : \varphi$ *such that:*

(i) for each trace value $\tau = x : \varphi$*, every path from the conclusion to an open leaf is covered by a trace* $[\alpha]\tau, \ldots, \tau$*; and*

(ii) every infinite path is followed by an infinitely progressing trace.



(a) Derivation schema for Axiom (1)

(b) Derivation schema for Axiom (2)

(c) Derivation schema for Axiom (7)

Fig. 5: **G3PDL**$^\omega$ derivation schemata for the distribution and induction axioms.

Schemas for deriving axioms (1), (2) and (7) are shown in fig. 5. Any infinite paths which exist in the schemas for deriving axioms (1) and (2) are followed by infinitely progressing traces by lemma 3. Thus, they are **G3PDL**$^\omega$ proofs. In the schema for axiom (7), the open leaves of the subderivation constructed via lemma 3 are converted into buds, the companion of each of which is the conclusion of the instance of the (∗R) rule. Condition (i) of lemma 3 guarantees that each infinite path along these cycles has an infinitely progressing trace. We thus have the following completeness result.

Theorem 3. *If φ is valid then $\Rightarrow x : \varphi$ is derivable in* **G3PDL**$^\omega$.

It should be noted that theorem 3 is not a deductive completeness result, i.e. it does not say that any sequent $\Gamma \Rightarrow \Delta$ is only valid if there is a **G3PDL**$^\omega$ proof for it. This is no major restriction, as a finitary syntactic consequence relation cannot capture semantic consequence in PDL: due to the presence of iteration, PDL is not compact. This can only be rectified by allowing infinite sequents in the proof system, which yields a system that is not amenable to automation.

5 Proof Search for Test-free, Acyclic Sequents

In this section, we describe a cut-free proof-search procedure for sequents containing formulas without tests (i.e. programs of the form $\varphi?$), and for which the relational atoms in the antecedents do not entail cyclic models.

Our approach relies on the following notion of normal form for sequents. For a set of relational atoms and labelled formulas, we write $*\text{-}\mathsf{labs}(\Gamma)$ for the set $\{x \mid x : [\alpha^*]\varphi \in \Gamma\}$. We call formulas of the form $[a]\varphi$ *basic*, those of the form $[\alpha^*]\varphi$ *iterated*, and the remaining non-atomic formulas *composite*.

Definition 12 (Normal Sequents). *A sequent $\Gamma \Rightarrow \Delta$ is called* normal *when: (1) $\Gamma \cap \Delta = \emptyset$; (2) Δ contains only labelled atomic and iterated formulas; and (3) Γ contains only relational atoms, labelled atomic formulas, and labelled basic formulas $x : [a]\varphi$ for which there is no y such that also $x\ R_a\ y \in \Gamma$.*

We say that x *reaches* y (or y is *reachable* from x) in Γ when there are labels $z_1, \ldots, z_n$ and atomic programs $a_1, \ldots, a_{n-1}$ such that $x = z_1$ and $y = z_n$ with $z_i\ R_{a_i}\ z_{n+1} \in \Gamma$ for each $i < n$. We say that a sequent $\Gamma \Rightarrow \Delta$ is *cyclic* if there is some $x \in \mathsf{labs}(\Gamma)$ such that x reaches itself in Γ; otherwise it is called *acyclic*.

Crucially, the following forms of weakening are validity-preserving.

Lemma 4 (Validity-preserving Weakenings). *The following hold.*
(1) If $\Gamma \Rightarrow \Delta, x\ R_a\ z$ is valid and $x\ R_a\ z \notin \Gamma$, then $\Gamma \Rightarrow \Delta$ is valid.
(2) If normal $\Gamma \Rightarrow \Delta, x : p$ is valid with $x \notin *\text{-}\mathsf{labs}(\Delta)$, then $\Gamma \Rightarrow \Delta$ is valid.
(3) If normal $\Gamma, x : \varphi \Rightarrow \Delta$ is valid with $x \notin \mathsf{labs}(\Delta)$, then $\Gamma \Rightarrow \Delta$ is valid.
(4) If normal $\Gamma, x\ R_a\ y \Rightarrow \Delta$ is valid, $z \in \mathsf{labs}(\Delta)$ for all $z : \varphi \in \Gamma$, $x \notin \mathsf{labs}(\Delta)$ and x not reachable in Γ from any $z \in \mathsf{labs}(\Delta)$, then $\Gamma \Rightarrow \Delta$ is valid.

An *unwinding* of a sequent $\Gamma \Rightarrow \Delta$ is a possibly open derivation of $\Gamma \Rightarrow \Delta$ obtained by applying left and right logical rules as much as possible, and satisfying the properties that: no trace progresses more than once; and all rule instances consume the active labelled formula of their conclusion, but preserve in the premise any relational atoms. A *capped* unwinding is an unwinding for which: (a) weakening rules and (Ax) and ($\bot$) have been applied to all open leaves $\Gamma \Rightarrow \Delta$ with $\bot \in \Gamma$ or $\Gamma \cap \Delta \neq \emptyset$; and (b) the sequence of weakenings in lemma 4 have been exhaustively applied to all other open leaves.

Lemma 5. *Let $\mathcal{D}$ be a capped unwinding for $\Gamma \Rightarrow \Delta$ (denoted S) and $\Gamma' \Rightarrow \Delta'$ an open leaf (denoted S') of $\mathcal{D}$. The following hold: (1) $\Gamma' \Rightarrow \Delta'$ is normal; (2) if $\Gamma \Rightarrow \Delta$ is valid, then so are all the open leaves of $\mathcal{D}$; and (3) For every trace $\boldsymbol{\tau}_n$ covering the path from S to S', if $\tau_1 = (x, \varepsilon, \beta, \varphi)$ is a sub-formula of τ_n, then the trace is progressing.*

We call a sequent *test-free* if it does not contain any programs of the form $\varphi?$. A crucial property for termination of the proof-search is the following.

Lemma 6. *Let $\mathcal{D}$ be a capped unwinding for a test-free, acyclic sequent; then $\mathcal{D}$ is finite, and $\mathsf{labs}(\Gamma') \subseteq \mathsf{labs}(\Delta') \subseteq *\text{-}\mathsf{labs}(\Delta')$ for all open leaves $\Gamma' \Rightarrow \Delta'$ of $\mathcal{D}$.*

Both cyclicity and the presence of tests can cause lemma 6 to fail, since then it is possible for there to be a path of ancestry between two occurrences of an antecedent formula $x : [\alpha^*]\varphi$ that traverses an instance of the ($*$L) rule. That is, antecedent formulas may be infinitely unfolded. Moreover, in the presence of tests or cyclicity, the weakenings of lemma 4(4) do not result in $\mathsf{labs}(\Gamma') \subseteq \mathsf{labs}(\Delta')$ for open leaves $\Gamma' \Rightarrow \Delta'$.

We define a function $*\text{-}\mathsf{max}$ on test-free sequents (details are given in the appendix), whose purpose is to provide a bound ensuring termination of proof-search. We have conjectured that it satisfies the following property.

Conjecture 1. Let $\mathcal{D}$ be a capped unwinding of test-free, acyclic $\Gamma \Rightarrow \Delta$. Then:
1. $|\{x : \varphi \in \Delta' \mid \varphi \text{ non-atomic}\}| \leq *\text{-}\mathsf{max}(\Gamma \Rightarrow \Delta)$.
2. $*\text{-}\mathsf{max}(\Gamma' \Rightarrow \Delta') \leq *\text{-}\mathsf{max}(\Gamma \Rightarrow \Delta)$ for all open leaves $\Gamma' \Rightarrow \Delta'$ of $\mathcal{D}$.

Proof-search proceeds by iteratively building capped unwindings for open leaves. All formulas encountered in the search are in the (finite) Fischer-Ladner closure of the initial sequent, and validity and acyclicity are preserved throughout the procedure. Lemma 6 and Conjecture 1 will ensure that the number of distinct open leaves (modulo relabelling) encountered during proof-search is bounded, so we may apply substitutions to form back-links during proof-search. Lemma 5(3) ensures that the resulting pre-proof satisfies the global trace condition. For invalid sequents, proof-search produces atomic sequents that are not axioms. We thus conjecture cut-free regular completeness for test-free PDL.

6 Conclusion

In this paper we have given two new non-wellfounded proof systems for PDL. $\mathbf{G3PDL}^\infty$ allows proof trees to be infinitely tall, and $\mathbf{G3PDL}^\omega$ restricts to the proofs of $\mathbf{G3PDL}^\infty$ that are finitely representable as cyclic graphs satisfying a trace condition. Soundness and completeness of both systems was shown, in particular, cut-free completeness of $\mathbf{G3PDL}^\infty$ and a strategy for cut-free completeness of $\mathbf{G3PDL}^\omega$ for test-free PDL.

There is much further work to be done. Of immediate interest is the verification of cut-free regular completeness for test-free PDL, and the extension of the argument to the full logic. We would also like to consider *additional* program constructs. Some, like converse, can already be treated through De Giacomo's [17] efficient translation of Converse PDL into PDL. It may be more desirable, however, to represent the program construct directly, to aid in the modular combination of different constructs. One construct that is particularly notorious is Intersection. Despite the modal definability of its dual, Choice, the intended interpretation of Intersection is *not* modally definable, and the completeness (and existence) of an axiomatisation for it remained open until Balbiani and Vakarelov [3]. An earlier, and significantly simpler, solution to this problem was the augmentation of PDL with nominals, denoted Combinatory DL [32]. We conjecture that the presence of labels in our system enables us to perform a similar trick, without contaminating the syntax of the logic itself. However we should note

that a key prerequisite of our soundness proof, namely that we can restrict attention to finitely branching models (guaranteed by the finite model property of PDL), is an assumption that may no longer hold for particular combinations of program constructs. Weakening this assumption will aid in the goal of giving a truly uniform proof theory for PDL-type logics.

Our work should be seen as a part of a wider program of research to give a uniform and modular proof theory for a larger group of modal logics, including what we have denoted PDL-type logics. One source of modularity and uniformity is the existing Negri labelled system our calculi extend. This allows us to freely add proof rules corresponding to first-order frame axioms defining Kripke models. A wider class of modal logics than those directly covered by Negri's framework are those with accessibility relations that are defined to be wellfounded or arise as transitive closures of other accessibility relations (we note Negri is able to treat the specific case of Gödel-Löb logic due to its special interpretation of $\Box$, but not the general class we describe). We believe an appropriate framework to uniformly capture *these* logics as well is cyclic labelled deduction. We are encouraged in this pursuit by recent work of Cohen and Rowe [8] in which first-order logic with a transitive closure operator is given a cyclic proof theory. We may think of labelled deduction as a way of giving a proof theoretic analysis of the first-order theory of Kripke models and their modal satisfaction relations. Labelled cyclic deduction, we conjecture, can capture the first-order-with-least-fixpoint theory of Kripke models and modal satisfaction relations.

Finally, and somewhat more speculatively, with the cyclic system in hand we intend to investigate the hitherto open problem of interpolation for PDL. This has seen no satisfactory resolution in the years since PDL was first formulated, with the only attempted proofs strongly disputed [27] or withdrawn [26]. It would be interesting to see if the existence of a straightforward proof system for the logic opens up any new lines of attack on the problem. For example, Lyndon interpolation has been proved for Gödel-Löb logic using a cyclic system [37].

References

1. Loredana Afanasiev, Patrick Blackburn, Ioanna Dimitriou, Bertrand Gaiffe, Evan Goris, Maarten Marx, and Maarten de Rijke. PDL for ordered trees. *Journal of Applied Non-Classical Logics*, 15(2):115–135, 2005.
2. Luca Alberucci. *The Modal μ-calculus and the Logic of Common Knowledge*. PhD thesis, Universität Bern, 2002.
3. Philippe Balbiani and Dimiter Vakarelov. PDL with intersection of programs: a complete axiomatization. *Journal of Applied Non-Classical Logics*, 13(3-4):231–276, 2003.
4. Johan Van Benthem. Modal frame correspondences and fixed-points. *Studia Logica*, 83(1):133–155, Jun 2006.
5. James Brotherston, Nikos Gorogiannis, and Rasmus L. Petersen. A generic cyclic theorem prover. In Ranjit Jhala and Atsushi Igarashi, editors, *Programming Languages and Systems*, pages 350–367, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

6. James Brotherston and Alex Simpson. Sequent calculi for induction and infinite descent. *J. Log. Comput.*, 21(6):1177–1216, 2011.
7. Facundo Carreiro and Yde Venema. PDL inside the μ-calculus: A syntactic and an automata-theoretic characterization. In Rajeev Goré, Barteld Kooi, and Agi Kurucz, editors, *Advances in Modal Logic, Volume 10*, pages 74–93. CSLI Publications, 2014.
8. Liron Cohen and Reuben N. S. Rowe. Uniform inductive reasoning in transitive closure logic via infinite descent. In *27th EACSL Annual Conference on Computer Science Logic, CSL 2018, September 4-7, 2018, Birmingham, UK*, pages 17:1–17:16, 2018.
9. Anupam Das and Damien Pous. A cut-free cyclic proof system for Kleene algebra. In *Automated Reasoning with Analytic Tableaux and Related Methods - 26th International Conference, TABLEAUX 2017, Brasília, Brazil, September 25-28, 2017, Proceedings*, pages 261–277, 2017.
10. Anupam Das and Damien Pous. Non-wellfounded proof theory for (Kleene+action)(algebras+lattices). In *27th EACSL Annual Conference on Computer Science Logic, CSL 2018, September 4-7, 2018, Birmingham, UK*, pages 19:1–19:18, 2018.
11. Giuseppe De Giacomo and Maurizio Lenzerini. Boosting the correspondence between description logics and propositional dynamic logics. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (Vol. 1)*, AAAI '94, pages 205–212, Menlo Park, CA, USA, 1994. American Association for Artificial Intelligence.
12. Nachum Dershowitz and Zohar Manna. Proving termination with multiset orderings. *Commun. ACM*, 22(8):465–476, August 1979.
13. Simon Docherty and Reuben N. S. Rowe. A non-wellfounded, labelled proof system for propositional dynamic logic. *CoRR*, abs/1905.06143, 2019.
14. Michael J. Fischer and Richard E. Ladner. Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences*, 18(2):194 – 211, 1979.
15. Melvin Fitting. *Proof Methods for Modal and Intuitionistic Logics*. Synthese Library. Springer Netherlands, 1983.
16. Sabine Frittella, Giuseppe Greco, Alexander Kurz, and Alessandra Palmigiano. Multi-type display calculus for propositional dynamic logic. *Journal of Logic and Computation*, 26(6):2067–2104, 11 2014.
17. Giuseppe De Giacomo. Eliminating "converse" from converse PDL. *Journal of Logic, Language, and Information*, 5(2):193–208, 1996.
18. Giuseppe De Giacomo and Fabio Massacci. Combining deduction and model checking into tableaux and algorithms for converse-PDL. *Information and Computation*, 162(1):117 – 137, 2000.
19. Patrick Girard, Jeremy Seligman, and Fenrong Liu. General dynamic dynamic logic. In Thomas Bolander, Torben Braüner, Silvio Ghilardi, and Lawrence Moss, editors, *Advances in Modal Logic, Volume 9*, pages 239–260. CSLI Publications, 2012.
20. Rajeev Goré and Florian Widmann. An optimal on-the-fly tableau-based decision procedure for PDL-satisfiability. In Renate A. Schmidt, editor, *Automated Deduction – CADE-22*, pages 437–452, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.
21. Jeroen Groenendijk and Martin Stokhof. Dynamic predicate logic. *Linguistics and Philosophy*, 14(1):39–100, Feb 1991.

22. David Harel. Dynamic logic. In Dov. Gabbay and Franz. Guenthner, editors, *Handbook of Philosophical Logic: Volume II: Extensions of Classical Logic*, pages 497–604. Springer Netherlands, Dordrecht, 1984.
23. David Harel, Jerzy Tiuryn, and Dexter Kozen. *Dynamic Logic*. MIT Press, Cambridge, MA, USA, 2000.
24. Brian Hill and Francesca Poggiolesi. A contraction-free and cut-free sequent calculus for propositional dynamic logic. *Studia Logica: An International Journal for Symbolic Logic*, 94(1):47–72, 2010.
25. Stig Kanger. *Provability in Logic*. Almqvist & Wiksell, 1957.
26. Tomasz Kowalski. Retraction note for "PDL has interpolation". *Journal of Symbolic Logic*, 69(3):935–936, 2004.
27. Marcus Kracht. Dynamic logic. In *Tools and Techniques in Modal Logic*, volume 142 of *Studies in Logic and the Foundations of Mathematics*, pages 497 – 533. Elsevier, 1999.
28. Sara Negri. Proof analysis in modal logic. *Journal of Philosophical Logic*, 34(5):507, Oct 2005.
29. Sara Negri. Proof theory for modal logic. *Philosophy Compass*, 6(8):523–538, 2011.
30. Damian Niwiński and Igor Walukiewicz. Games for the μ-calculus. *Theoretical Computer Science*, 163(1):99 – 116, 1996.
31. Rohit Parikh. Propositional game logic. In *Proceedings of the 24th Annual Symposium on Foundations of Computer Science*, SFCS '83, pages 195–200, Washington, DC, USA, 1983. IEEE Computer Society.
32. Solomon Passy and Tinko Tinchev. An essay in combinatory dynamic logic. *Information and Computation*, 93(2):263 – 332, 1991.
33. André Platzer. Differential dynamic logic for hybrid systems. *Journal of Automated Reasoning*, 41(2):143–189, Aug 2008.
34. Vaughan R. Pratt. Semantical consideration on Floyd-Hoare logic. In *17th Annual Symposium on Foundations of Computer Science (SFCS 1976)*, pages 109–121, Oct 1976.
35. Gerard Renardel de Lavalette, Barteld Kooi, and Rineke Verbrugge. Strong completeness and limited canonicity for PDL. *Journal of Logic, Language and Information*, 17(1):69–87, Jan 2008.
36. Reuben N. S. Rowe and James Brotherston. Automatic cyclic termination proofs for recursive procedures in separation logic. In *Proceedings of the 6th ACM SIGPLAN Conference on Certified Programs and Proofs, CPP 2017, Paris, France, January 16-17, 2017*, pages 53–65, 2017.
37. D. S. Shamkanov. Circular proofs for the Gödel-Löb provability logic. *Mathematical Notes*, 96(3):575–585, Sep 2014.
38. Gadi Tellez and James Brotherston. Automatically verifying temporal properties of pointer programs with cyclic proof. In *Automated Deduction - CADE 26 - 26th International Conference on Automated Deduction, Gothenburg, Sweden, August 6-11, 2017, Proceedings*, pages 491–508, 2017.
39. Albert Visser. *Löb's Logic Meets the μ-Calculus*, pages 14–25. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.
40. Ricardo Wehbe. *Annotated systems for common knowledge*. PhD thesis, Universität Bern, 2010.