Cyclic Proofs, Hypersequents, and Transitive Closure Logic

Das, Anupam; Girlando, Marianna

DOI:
10.1007/978-3-031-10769-6_30

Cyclic Proofs, Hypersequents, and Transitive Closure Logic

Anupam Das(✉) and Marianna Girlando

University of Birmingham, Birmingham, UK
{a.das,m.girlando}@bham.ac.uk

Abstract. We propose a cut-free cyclic system for Transitive Closure Logic (TCL) based on a form of *hypersequents*, suitable for automated reasoning via proof search. We show that previously proposed sequent systems are cut-free incomplete for basic validities from Kleene Algebra (KA) and Propositional Dynamic Logic (PDL), over standard translations. On the other hand, our system faithfully simulates known cyclic systems for KA and PDL, thereby inheriting their completeness results. A peculiarity of our system is its richer correctness criterion, exhibiting 'alternating traces' and necessitating a more intricate soundness argument than for traditional cyclic proofs.

Keywords: Cyclic proofs · Transitive Closure Logic · Hypersequents · Propositional Dynamic Logic

1 Introduction

Transitive Closure Logic (TCL) is the extension of first-order logic by an operator computing the transitive closure of definable binary relations. It has been studied by numerous authors, e.g. [15–17], and in particular has been proposed as a foundation for the mechanisation and automation of mathematics [1].

Recently, Cohen and Rowe have proposed *non-wellfounded* and *cyclic* systems for TCL [9,11]. These systems differ from usual ones by allowing proofs to be infinite (finitely branching) trees, rather than finite ones, under some appropriate global correctness condition (the 'progressing criterion'). One particular feature of the cyclic approach to proof theory is the facilitation of automation, since complexity of inductive invariants is effectively traded off for a richer proof structure. In fact this trade off has recently been made formal, cf. [3,12], and has led to successful applications to automated reasoning, e.g. [6,7,24,26,27].

In this work we investigate the capacity of cyclic systems to automate reasoning in TCL. Our starting point is the demonstration of a key shortfall of Cohen and Rowe's system: its cut-free fragment, here called TC_G, is unable to cyclically prove even standard theorems of relational algebra, e.g. $(a \cup b)^* = a^*(ba^*)^*$ and

This work was supported by a UKRI Future Leaders Fellowship, 'Structure vs Invariants in Proofs', project reference MR/S035540/1.

J. Blanchette et al. (Eds.): IJCAR 2022, LNAI 13385, pp. 509–528, 2022.
https://doi.org/10.1007/978-3-031-10769-6_30

$(aa \cup aba)^+ \leq a^+((ba^+)^+ \cup a))$ (Theorem 12). An immediate consequence of this is that cyclic proofs of TC_G do not enjoy cut-admissibility (Corollary 13). On the other hand, these (in)equations are theorems of Kleene Algebra (KA) [18,19], a decidable theory which admits automation-via-proof-search thanks to the recent cyclic system of Das and Pous [14].

What is more, TCL is well-known to interpret Propositional Dynamic Logic (PDL), a modal logic whose modalities are just terms of KA, by a natural extension of the 'standard translation' from (multi)modal logic to first-order logic (see, e.g., [4,5]). Incompleteness of cyclic-TC_G for PDL over this translation is inherited from its incompleteness for KA. This is in stark contrast to the situation for modal logics without fixed points: the standard translation from K (and, indeed, all logics in the 'modal cube') to first-order logic actually *lifts* to cut-free proofs for a wide range of modal logic systems, cf. [21,22].

A closer inspection of the systems for KA and PDL reveals the stumbling block to any simulation: these systems implicitly conduct a form of 'deep inference', by essentially reasoning underneath $\exists$ and $\wedge$. Inspired by this observation, we propose a form of *hypersequents* for predicate logic, with extra structure admitting the deep reasoning required. We present the cut-free system HTC and a novel notion of cyclic proof for these hypersequents. In particular, the incorporation of some deep inference at the level of the rules necessitates an 'alternating' trace condition corresponding to *alternation* in automata theory.

Our first main result is the Soundness Theorem (Theorem 23): non-wellfounded proofs of HTC are sound for *standard semantics*. The proof is rather more involved than usual soundness arguments in cyclic proof theory, due to the richer structure of hypersequents and the corresponding progress criterion. Our second main result is the Simulation Theorem (Theorem 28): HTC is complete for PDL over the standard translation, by simulating a cut-free cyclic system for the latter. This result can be seen as a formal interpretation of cyclic modal proof theory within cyclic predicate proof theory, in the spirit of [21,22].

To simplify the exposition, we shall mostly focus on equality-free TCL and 'identity-free' PDL in this paper, though all our results hold also for the 'reflexive' extensions of both logics. We discuss these extensions in Sect. 7, and present further insights and conclusions in Sect. 8. Full proofs and further examples not included here (due to space constraints) can be found in [13].

2 Preliminaries

We shall work with a fixed first-order vocabulary consisting of a countable set Pr of unary *predicate* symbols, written p, q, etc., and of a countable set Rel of binary *relation* symbols, written a, b, etc. We shall generally reserve the word 'predicate' for unary and 'relation' for binary. We could include further relational symbols too, of higher arity, but choose not to in order to calibrate the semantics of both our modal and predicate settings.

We build formulas from this language differently in the modal and predicate settings, but all our formulas may be formally evaluated within *structures*:

Definition 1 (Structures). *A* structure $\mathcal{M}$ *consists of a set* D*, called the* domain *of* $\mathcal{M}$*, which we sometimes denote by* $|\mathcal{M}|$*; a subset* $p^{\mathcal{M}} \subseteq D$ *for each* $p \in \mathsf{Pr}$*; and a subset* $a^{\mathcal{M}} \subseteq D \times D$ *for each* $a \in \mathsf{Rel}$*.*

2.1 Transitive Closure Logic

In addition to the language introduced at the beginning of this section, in the predicate setting we further make use of a countable set of *function* symbols, written f^i, g^j, etc. where the superscripts $i, j \in \mathbb{N}$ indicate the *arity* of the function symbol and may be omitted when it is not ambiguous. Nullary function symbols (aka *constant* symbols), are written c, d etc. We shall also make use of *variables*, written x, y, etc., typically bound by quantifiers. *Terms*, written s, t, etc., are generated as usual from variables and function symbols by function application. A term is *closed* if it has no variables.

We consider the usual syntax for first-order logic formulas over our language, with an additional operator for transitive closure (and its dual). Formally, TCL formulas, written A, B, etc., are generated as follows:

$$\begin{aligned} A, B ::= {} & p(t) \mid \bar{p}(t) \mid a(s,t) \mid \bar{a}(s,t) \mid (A \wedge B) \mid (A \vee B) \mid \forall x A \mid \exists x A \mid \\ & TC(\lambda x, y.A)(s,t) \mid \overline{TC}(\lambda x, y.A)(s,t) \end{aligned}$$

When variables x, y are clear from context, we may write $TC(A(x,y))(s,t)$ or $TC(A)(s,t)$ instead of $TC(\lambda x, y.A)(s,t)$, as an abuse of notation, and similarly for $\overline{TC}$. We may write $A[t/x]$ for the formula obtained from A by replacing every free occurrence of the variable x by the term t. We have included both TC and $\overline{TC}$ as primitive operators, so that we can reduce negation to atomic formulas, shown below. This will eventually allow a one-sided formulation of proofs.

Definition 2 (Duality). *For a formula* A *we define its* complement*,* $\bar{A}$*, by:*

$$\begin{array}{lllllll} \overline{p(t)} & := \bar{p}(t) & \overline{\bar{p}(t)} & := p(t) & \overline{A \wedge B} & := \bar{A} \vee \bar{B} & \overline{TC(A)(s,t)} := \overline{TC}(\bar{A})(s,t) \\ \overline{a(s,t)} & := \bar{a}(s,t) & \overline{\forall x A} & := \exists x \bar{A} & \overline{A \vee B} & := \bar{A} \wedge \bar{B} & \overline{\overline{TC}(A)(s,t)} := TC(\bar{A})(s,t) \\ \overline{\bar{a}(s,t)} & := a(s,t) & \overline{\exists x A} & := \forall x \bar{A} & & & \end{array}$$

We shall employ standard logical abbreviations, e.g. $A \supset B$ for $\bar{A} \vee B$.

We may evaluate formulas with respect to a structure, but we need additional data for interpreting function symbols:

Definition 3 (Interpreting function symbols). *Let* $\mathcal{M}$ *be a structure with domain* D*. An* interpretation *is a map* ρ *that assigns to each function symbol* f^n *a function* $D^n \to D$*. We may extend any interpretation* ρ *to an action on (closed) terms by setting recursively* $\rho(f(t_1, \ldots, t_n)) := \rho(f)(\rho(t_1), \ldots, \rho(t_n))$*.*

We only consider *standard* semantics in this work: TC (and $\overline{TC}$) is always interpreted as the *real* transitive closure (and its dual) in a structure, rather than being axiomatised by some induction (and coinduction) principle.

Definition 4 (Semantics). *Given a structure $\mathcal{M}$ with domain D and an interpretation ρ, the judgement $\mathcal{M}, \rho \models A$ is defined as usual for first-order logic with the following additional clauses for TC and $\overline{TC}$:*[1]

- $\mathcal{M}, \rho \models TC(A(x,y))(s,t)$ *if there are* $v_0, \ldots, v_{n+1} \in D$ *with* $\rho(s) = v_0$, $\rho(t) = v_{n+1}$, *such that for every* $i \leq n$ *we have* $\mathcal{M}, \rho \models A(v_i, v_{i+1})$.
- $\mathcal{M}, \rho \models \overline{TC}(A(x,y))(s,t)$ *if for all* $v_0, \ldots, v_{n+1} \in D$ *with* $\rho(s) = v_0$ *and* $\rho(t) = v_{n+1}$, *there is some* $i \leq n$ *such that* $\mathcal{M}, \rho \models A(v_i, v_{i+1})$.

If $\mathcal{M}, \rho \models A$ for all $\mathcal{M}$ and ρ, we simply write $\models A$.

Remark 5 (TC and $\overline{TC}$ as least and greatest fixed points). As expected, we have $\mathcal{M}, \rho \not\models TC(A)(s,t)$ just if $\mathcal{M}, \rho \models \overline{TC}(\bar{A})(s,t)$, and so the two operators are semantically dual. Thus, TC and $\overline{TC}$ duly correspond to least and greatest fixed points, respectively, satisfying in any model:

$$TC(A)(s,t) \iff A(s,t) \vee \exists x(A(s,x) \wedge TC(A)(x,t)) \tag{1}$$

$$\overline{TC}(A)(s,t) \iff A(s,t) \wedge \forall x(A(s,x) \vee \overline{TC}(A)(x,t)) \tag{2}$$

Let us point out that our $\overline{TC}$ operator is not the same as Cohen and Rowe's transitive 'co-closure' operator TC^{op} in [10], but rather the De Morgan dual of TC. In the presence of negation, TC and $\overline{TC}$ are indeed interdefinable, cf. Definition 2.

2.2 Cohen-Rowe Cyclic System for TCL

Cohen and Rowe proposed in [9,11] a non-wellfounded system for TCL that extends a usual sequent calculus $\mathsf{LK}_=$ for first-order logic with equality and substitution by rules for TC inspired by its characterisation as a least fixed point, cf. (1).[2] Note that the presence of the substitution rule is critical for the notion of 'regularity' in predicate cyclic proof theory. The resulting notions of non-wellfounded and cyclic proofs are formulated similarly to those for first-order logic with (ordinary) inductive definitions [8]:

Definition 6 (Sequent system). TC_G *is the extension of* $\mathsf{LK}_=$ *by the rules:*

$$\begin{array}{c} {}_{TC_0}\dfrac{\Gamma, A(s,t)}{\Gamma, TC(A)(s,t)} \qquad {}_{TC_1}\dfrac{\Gamma, A(s,r) \quad \Gamma, TC(A)(r,t)}{\Gamma, TC(A)(s,t)} \\ {}_{\overline{TC}}\dfrac{\Gamma, A(s,t) \quad \Gamma, A(s,c), \overline{TC}(A)(c,t)}{\Gamma, \overline{TC}(A)(s,t)}\ c \text{ fresh} \end{array} \tag{3}$$

TC_G-preproofs *are possibly infinite trees of sequents generated by the rules of* TC_G. *A preproof is* regular *if it has only finitely many distinct sub-preproofs.*

[1] Note that we are including 'parameters from the model' in formulas here. Formally, this means each $v \in D$ is construed as a constant symbol for which $\rho(v) = v$.

[2] Cohen and Rowe's system is originally called RTC_G, rather using a 'reflexive' version RTC of the TC operator. However this (and its rules) can be encoded (and simulated) by defining $RTC(\lambda x, y.A)(s,t) := TC(\lambda x, y(x = y \vee A))(s,t)$.

The notion of 'correct' non-wellfounded proof is obtained by a standard *progressing criterion* in cyclic proof theory. We shall not go into details here, being beyond the scope of this work, but refer the reader to those original works (as well as [13] for our current variant). Let us write $\vdash_{cyc}$ for their notion of cyclic provability using the above rules, cf. [9,11]. A standard infinite descent counter-model argument yields:

Proposition 7 (Soundness, [9,11]). *If* $\mathsf{TC}_G \vdash_{cyc} A$ *then* $\models A$.

In fact, this result is subsumed by our main soundness result for HTC (Theorem 23) and its simulation of TC_G (Theorem 19). In the presence of cut, a form of converse of Proposition 7 holds: cyclic TC_G proofs are 'Henkin complete', i.e. complete for all models of a particular axiomatisation of TCL based on (co)induction principles for TC (and $\overline{TC}$) [9,11]. However, the counterexample we present in the next section implies that cut is not eliminable (Corollary 13).

3 Interlude: Motivation from PDL and Kleene Algebra

Given the TCL sequent system proposed by Cohen and Rowe, why do we propose a hypersequential system? Our main argument is that proof search in TC_G is rather weak, to the extent that cut-free cyclic proofs are unable to simulate a basic (cut-free) system for modal logic PDL (regardless of proof search strategy). At least one motivation here is to 'lift' the *standard translation* from cut-free cyclic proofs for PDL to cut-free cyclic proofs in an adequate system for TCL.

3.1 Identity-Free PDL

Identity-free propositional dynamic logic (PDL^+) is a version of the modal logic PDL without tests or identity, thereby admitting an 'equality-free' standard translation into predicate logic. Formally, PDL^+ *formulas*, written A, B, etc., and *programs*, written α, β, etc., are generated by the following grammars:

$$A, B ::= p \mid \overline{p} \mid (A \wedge B) \mid (A \vee B) \mid [\alpha]A \mid \langle\alpha\rangle A$$
$$\alpha, \beta ::= a \mid (\alpha;\beta) \mid (\alpha \cup \beta) \mid \alpha^+$$

We sometimes simply write $\alpha\beta$ instead of $\alpha;\beta$, and $(\alpha)A$ for a formula that is either $\langle\alpha\rangle A$ or $[\alpha]A$.

Definition 8 (Duality). *For a formula A we define its* complement*, $\bar{A}$, by:*

$$\bar{\overline{p}} := p \qquad \begin{array}{ll} \overline{A \wedge B} := \bar{A} \vee \bar{B} & \overline{[\alpha]A} := \langle\alpha\rangle\bar{A} \\ \overline{A \vee B} := \bar{A} \wedge \bar{B} & \overline{\langle\alpha\rangle A} := [\alpha]\bar{A} \end{array}$$

We *evaluate* PDL^+ formulas using the traditional relational semantics of modal logic, by associating each program with a binary relation in a structure. Again, we only consider standard semantics, in the sense that the $+$ operator is interpreted as the real transitive closure within a structure.

Definition 9 (Semantics). *For structures $\mathcal{M}$ with domain D, elements $v \in D$, programs α and formulas A, we define $\alpha^{\mathcal{M}} \subseteq D \times D$ and the judgement $\mathcal{M}, v \models A$ as follows:*

- *($a^{\mathcal{M}}$ is already given in the specification of $\mathcal{M}$, cf. Definition 1).*
- $(\alpha;\beta)^{\mathcal{M}} := \{(u,v) : \text{there is } w \in D \text{ s.t. } (u,w) \in \alpha^{\mathcal{M}} \text{ and } (w,v) \in \beta^{\mathcal{M}}\}$.
- $(\alpha \cup \beta)^{\mathcal{M}} := \{(u,v) : (u,v) \in \alpha^{\mathcal{M}} \text{ or } (u,v) \in \beta^{\mathcal{M}}\}$.
- $(\alpha^+)^{\mathcal{M}} := \{(u,v) : \text{there are } w_0, \ldots, w_{n+1} \in D \text{ s.t. } u = w_0, v = w_{n+1} \text{ and, for every } i \leq n, (w_i, w_{i+1}) \in \alpha^{\mathcal{M}}\}$.

- $\mathcal{M}, v \models p$ *if* $v \in p^{\mathcal{M}}$.
- $\mathcal{M}, v \models \overline{p}$ *if* $v \notin p^{\mathcal{M}}$.
- $\mathcal{M}, v \models A \wedge B$ *if* $\mathcal{M}, v \models A$ *and* $\mathcal{M}, v \models B$.
- $\mathcal{M}, v \models A \vee B$ *if* $\mathcal{M}, v \models A$ *or* $\mathcal{M}, v \models B$.
- $\mathcal{M}, v \models [\alpha]A$ *if* $\forall\, (v,w) \in \alpha^{\mathcal{M}}$ *we have* $\mathcal{M}, w \models A$.
- $\mathcal{M}, v \models \langle\alpha\rangle A$ *if* $\exists\, (v,w) \in \alpha^{\mathcal{M}}$ *with* $\mathcal{M}, w \models A$.

If $\mathcal{M}, v \models A$ for all $\mathcal{M}$ and $v \in |\mathcal{M}|$, then we write $\models A$.

Note that we are overloading the satisfaction symbol $\models$ here, for both PDL$^+$ and TCL. This should never cause confusion, in particular since the two notions of satisfaction are 'compatible' as we shall now see.

3.2 The Standard Translation

The so-called 'standard translation' of modal logic into predicate logic is induced by reading the semantics of modal logic as first-order formulas. We now give a natural extension of this that interprets PDL$^+$ into TCL. At the logical level our translation coincides with the usual one for basic modal logic; our translation of programs, as expected, requires the TC operator to interpret the $+$ of PDL$^+$.

Definition 10. *For* PDL$^+$ *formulas A and programs α, we define the* standard translations $\mathsf{ST}(A)(x)$ *and* $\mathsf{ST}(\alpha)(x,y)$ *as* TCL*-formulas with free variables x and x, y, resp., inductively as follows:*

$$\begin{aligned}
\mathsf{ST}(p)(x) &:= p(x) & \mathsf{ST}(a)(x,y) &:= a(x,y)\\
\mathsf{ST}(\overline{p})(x) &:= \overline{p}(x) & \mathsf{ST}(\alpha \cup \beta)(x,y) &:= \mathsf{ST}(\alpha)(x,y) \vee \mathsf{ST}(\beta)(x,y)\\
\mathsf{ST}(A \vee B)(x) &:= \mathsf{ST}(A)(x) \vee \mathsf{ST}(B)(x) & \mathsf{ST}(\alpha;\beta)(x,y) &:= \exists z(\mathsf{ST}(\alpha)(x,z) \wedge \mathsf{ST}(\beta)(z,y))\\
\mathsf{ST}(A \wedge B)(x) &:= \mathsf{ST}(A)(x) \wedge \mathsf{ST}(B)(x) & \mathsf{ST}(\alpha^+)(x,y) &:= TC(\mathsf{ST}(\alpha))(x,y)\\
\mathsf{ST}(\langle\alpha\rangle A)(x) &:= \exists y(\mathsf{ST}(\alpha)(x,y) \wedge \mathsf{ST}(A)(y))\\
\mathsf{ST}([\alpha]A)(x) &:= \forall y(\overline{\mathsf{ST}(\alpha)(x,y)} \vee \mathsf{ST}(A)(y))
\end{aligned}$$

where $TC(\mathsf{ST}(\alpha))$ is shorthand for $TC(\lambda x, y.\mathsf{ST}(\alpha)(x,y))$.

It is routine to show that $\overline{\mathsf{ST}(A)(x)} = \mathsf{ST}(\bar{A})(x)$, by structural induction on A, justifying our overloading of the notation $\bar{A}$, in both TCL and PDL$^+$. Yet another advantage of using the same underlying language for both the modal and predicate settings is that we can state the following (expected) result without the need for encodings, following by a routine structural induction (see, e.g., [5]):

Theorem 11. *For* PDL$^+$ *formulas A, we have $\mathcal{M}, v \models A$ iff $\mathcal{M} \models \mathsf{ST}(A)(v)$.*

3.3 Cohen-Rowe System is not Complete for PDL$^+$

PDL$^+$ admits a standard cut-free cyclic proof system LPD^+ (see Sect. 6.1) which is both sound and complete (cf. Theorem 30). However, a shortfall of TC_G is that it is unable to cut-free simulate LPD^+. In fact, we can say something stronger:

Theorem 12 (Incompleteness). *There exist a* PDL$^+$ *formula A such that $\models A$ but $\mathsf{TC}_G \nvdash_{cyc} \mathsf{ST}(A)(x)$ (in the absence of cut).*

This means not only that TC_G is unable to locally cut-free simulate the rules of LPD^+, but also that there are some validities for which there are no cut-free cyclic proofs at all in TC_G. One example of such a formula is:

$$\langle (aa \cup aba)^+ \rangle p \supset \langle a^+((ba^+)^+ \cup a)\rangle p \tag{4}$$

A detailed proof of this is found in [13], but let us briefly discuss it here. First, the formula above is not artificial: it is derived from the well-known PDL validity $\langle (a \cup b)^* \rangle p \supset \langle a^*(ba^*)^* \rangle p$ by identity-elimination. This in turn is essentially a theorem of relational algebra, namely $(a \cup b)^* \leq a^*(ba^*)^*$, which is often used to eliminate $\cup$ in (sums of) regular expressions. The same equation was (one of those) used by Das and Pous in [14] to show that the sequent system LKA for Kleene Algebra is cut-free cyclic incomplete.

The argument that $\mathsf{TC}_G \nvdash_{cyc} \mathsf{ST}(4)(x)$ is much more involved than the one from [14], due to the fact we are working in predicate logic, but the underlying basic idea is similar. At a very high level, the RHS of (4) (viewed as a relational inequality) is translated to an existential formula $\exists z(\mathsf{ST}(a^+)(x,z) \wedge \mathsf{ST}((ba^+)^+ \cup a)(z,y)$ that, along some branch (namely the one that always chooses aa when decomposing the LHS of (4)) can never be instantiated while remaining valid. This branch witnesses the non-regularity of any proof. However $\mathsf{ST}(4)(x)$ is cyclically provable in TC_G with cut, so an immediate consequence of Theorem 12 is:

Corollary 13. *The class of cyclic proofs of TC_G does not enjoy cut-admissibility.*

4 Hypersequent Calculus for TCL

Let us take a moment to examine why any 'local' simulation of LPD^+ by TC_G fails, in order to motivate the main system that we shall present. The program rules, in particular the $\langle\,\rangle$-rules, require a form of *deep inference* to be correctly simulated, over the standard translation. For instance, let us consider the action of the standard translation on two rules we shall see later in LPD^+ (cf. Sect. 6.1):

$$\langle\cup\rangle_0\,\frac{\Gamma, \langle a_0\rangle p}{\Gamma, \langle a_0 \cup a_1\rangle p} \quad\rightsquigarrow\quad \frac{\mathsf{ST}(\Gamma)(c), \exists x(a_0(c,x) \wedge p(x))}{\mathsf{ST}(\Gamma)(c), \exists x((a_0(c,x) \vee a_1(c,x)) \wedge p(x))}$$

$$\langle;\rangle\,\frac{\Gamma, \langle a\rangle\langle b\rangle p}{\Gamma, \langle a;b\rangle p} \quad\rightsquigarrow\quad \frac{\mathsf{ST}(\Gamma)(c), \exists y(a(c,y) \wedge \exists x(b(y,x) \wedge p(x)))}{\mathsf{ST}(\Gamma)(c), \exists x(\exists y(a(c,y) \wedge b(y,x)) \wedge p(x))}$$

$$\mathsf{init}\,\frac{}{\{\,\}^{\varnothing}} \qquad \mathsf{wk}\,\frac{\mathbf{S}}{\mathbf{S},\mathbf{S}'} \qquad \sigma\,\frac{\mathbf{S}}{\sigma(\mathbf{S})} \qquad \cup\,\frac{\mathbf{S},\{\Gamma\}^{\mathbf{x}} \quad \mathbf{S},\{\Delta\}^{\mathbf{y}}}{\mathbf{S},\{\Gamma,\Delta\}^{\mathbf{x},\mathbf{y}}}\ \begin{array}{l}\mathsf{fv}(\Delta)\cap\mathbf{x}=\varnothing\\ \mathsf{fv}(\Gamma)\cap\mathbf{y}=\varnothing\end{array}$$

$$\mathsf{id}\,\frac{\mathbf{S},\{\Gamma\}^{\mathbf{x}}}{\mathbf{S},\{\Gamma,A\}^{\mathbf{x}},\{\overline{A}\}^{\varnothing}}\ A \text{ closed} \qquad \wedge\,\frac{\mathbf{S},\{\Gamma,A,B\}^{\mathbf{x}}}{\mathbf{S},\{\Gamma,A\wedge B\}^{\mathbf{x}}} \qquad \vee\,\frac{\mathbf{S},\{\Gamma,A_i\}^{\mathbf{x}}}{\mathbf{S},\{\Gamma,A_0\vee A_1\}^{\mathbf{x}}}\ i\in\{0,1\}$$

$$\mathsf{inst}\,\frac{\mathbf{S},\{\Gamma(t)\}^{\mathbf{x}}}{\mathbf{S},\{\Gamma(y)\}^{\mathbf{x},y}} \qquad \exists\,\frac{\mathbf{S},\{\Gamma,A(y)\}^{\mathbf{x},y}}{\mathbf{S},\{\Gamma,\exists x(A(x))\}^{\mathbf{x}}}\ y \text{ fresh} \qquad \forall\,\frac{\mathbf{S},\{\Gamma,A(f(\mathbf{x}))\}^{\mathbf{x}}}{\mathbf{S},\{\Gamma,\forall x(A(x))\}^{\mathbf{x}}}\ f \text{ fresh}$$

$$TC\,\frac{\mathbf{S},\{\Gamma,A(s,t)\}^{\mathbf{x}},\{\Gamma,A(s,z),TC(A)(z,t)\}^{\mathbf{x},z}}{\mathbf{S},\{\Gamma,TC(A)(s,t)\}^{\mathbf{x}}}\ z \text{ fresh}$$

$$\overline{TC}\,\frac{\mathbf{S},\{\Gamma,A(s,t),A(s,f(\mathbf{x}))\}^{\mathbf{x}},\{\Gamma,A(s,t),\overline{TC}(A)(f(\mathbf{x}),t)\}^{\mathbf{x}}}{\mathbf{S},\{\Gamma,\overline{TC}(A)(s,t)\}^{\mathbf{x}}}\ f \text{ fresh}$$

Fig. 1. Hypersequent calculus HTC. σ is a 'substitution' map from constants to terms and a renaming of other function symbols and variables.

The first case above suggests that any system to which the standard translation lifts must be able to reason *underneath* $\exists$ *and* $\wedge$, so that the inference indicated in blue is 'accessible' to the prover. The second case above suggests that the existential-conjunctive meta-structure necessitated by the first case should admit basic equivalences, in particular certain *prenexing*. This section is devoted to the incorporation of these ideas (and necessities) into a bona fide proof system.

4.1 A System for Predicate Logic via Annotated Hypersequents

An *annotated cedent*, or simply *cedent*, written S, S' etc., is an expression $\{\Gamma\}^{\mathbf{x}}$, where Γ is a set of formulas and the *annotation* $\mathbf{x}$ is a set of variables. We sometimes construe annotations as lists rather than sets when it is convenient, e.g. when taking them as inputs to a function.

Each cedent may be intuitively read as a TCL formula, under the following interpretation: $fm(\{\Gamma\}^{x_1,\ldots,x_n}) := \exists x_1 \ldots \exists x_n \bigwedge \Gamma$. When $\mathbf{x} = \varnothing$ then there are no existential quantifiers above, and when $\Gamma = \varnothing$ we simply identify $\bigwedge \Gamma$ with $\top$. We also sometimes write simply A for the annotated cedent $\{A\}^{\varnothing}$.

A *hypersequent*, written $\mathbf{S}, \mathbf{S}'$ etc., is a set of annotated cedents. Each hypersequent may be intuitively read as the disjunction of its cedents. Namely we set: $fm(\{\Gamma_1\}^{\mathbf{x}_1}, \ldots, \{\Gamma_n\}^{\mathbf{x}_n}) := fm(\{\Gamma_1\}^{\mathbf{x}_1}) \vee \ldots \vee fm(\{\Gamma_n\}^{\mathbf{x}_n})$.

Definition 14 (System). *The rules of* HTC *are given in Fig. 1. A* HTC preproof *is a (possibly infinite) derivation tree generated by the rules of* HTC. *A preproof is* regular *if it has only finitely many distinct subproofs.*

Our hypersequential system is somewhat more refined than usual sequent systems for predicate logic. E.g., the usual $\exists$ rule is decomposed into $\exists$ and inst,

whereas the usual $\wedge$ rule is decomposed into $\wedge$ and $\cup$. The rules for TC and $\overline{TC}$ are induced directly from their characterisations as fixed points in (1).

Note that the rules $\overline{TC}$ and $\forall$ introduce, bottom-up, the fresh function symbol f, which plays the role of the *Herbrand function* of the corresponding $\forall$ quantifier: just as $\forall \mathbf{x} \exists x A(x)$ is equisatisfiable with $\forall \mathbf{x} A(f(\mathbf{x}))$, when f is fresh, by Skolemisation, by duality $\exists \mathbf{x} \forall x A(x)$ is equivalid with $\exists \mathbf{x} A(f(\mathbf{x}))$, when f is fresh, by Herbrandisation. The usual $\forall$ rule of the sequent calculus corresponds to the case when $\mathbf{x} = \varnothing$.

4.2 Non-wellfounded Hypersequent Proofs

Our notion of ancestry, as compared to traditional sequent systems, must account for the richer structure of hypersequents:

Definition 15 (Ancestry). *Fix an inference step* r*, as typeset in Fig. 1. A formula* C *in the premiss is an* immediate ancestor *of a formula* C' *in the conclusion if they have the same colour; if* $C, C' \in \Gamma$ *then we further require* $C = C'$*, and if* C, C' *occur in* $\mathbf{S}$ *then* $C = C'$ *occur in the same cedent. A cedent* S *in the premiss is an immediate ancestor of a cedent* S' *in the conclusion if some formula in* S *is an* immediate ancestor *of some formula in* S'*.*

Immediate ancestry on both formulas and cedents is a binary relation, inducing a directed graph whose paths form the basis of our correctness condition:

Definition 16 ((Hyper)traces). *A* hypertrace *is a maximal path in the graph of immediate ancestry on cedents. A* trace *is a maximal path in the graph of immediate ancestry on formulas.*

Definition 17 (Progress and proofs). *Fix a preproof* $\mathcal{D}$*. A (infinite) trace* $(F_i)_{i \in \omega}$ *is* progressing *if there is* k *such that, for all* $i > k$*,* F_i *has the form* $\overline{TC}(A)(s_i, t_i)$ *and is infinitely often principal.*[3] *A (infinite) hypertrace* $\mathcal{H}$ *is* progressing *if every infinite trace within it is progressing. A (infinite) branch is progressing if it has a progressing hypertrace.* $\mathcal{D}$ *is a* proof *if every infinite branch is progressing. If, furthermore,* $\mathcal{D}$ *is regular, we call it a* cyclic proof*.*

We write $\mathsf{HTC} \vdash_{nwf} \mathbf{S}$ *(or* $\mathsf{HTC} \vdash_{cyc} \mathbf{S}$*) if there is a proof (or cyclic proof, respectively) of* HTC *of the hypersequent* $\mathbf{S}$*.*

In usual cyclic systems, checking that a regular preproof is progressing is decidable by straightforward reduction to the universality of nondeterministic ω-automata, with runs 'guessing' a progressing trace along an infinite branch. Our notion of progress exhibits an extra quantifier alternation: we must *guess* an infinite hypertrace in which *every* trace is progressing. Nonetheless, by appealing to determinisation or alternation, we can still decide our progressing condition:

Proposition 18. *Checking whether a* HTC *preproof is a proof is decidable by reduction to universality of* ω*-regular languages.*

[3] In fact, by a simple well-foundedness argument, it is equivalent to say that $(F_i)_{i<\omega}$ is progressing if it is infinitely often principal for a $\overline{TC}$-formula.

As we mentioned earlier, cyclic proofs of HTC indeed are at least as expressive as those of Cohen and Rowe's system by a routine local simulation of rules:

Theorem 19 (Simulating Cohen-Rowe). *If* $\mathsf{TC}_G \vdash_{cyc} A$ *then* $\mathsf{HTC} \vdash_{cyc} A$.

4.3 Some Examples

Example 20 (Fixed point identity). The sequent $\{\overline{TC}(a)(c,d)\}^{\varnothing}, \{TC(\bar{a})(c,d)\}^{\varnothing}$ is finitely derivable using rule id on $TC(a)(c,d)$ and the init rule. However we can also cyclically reduce it to a simpler instance of id. Due to the granularity of the inference rules of HTC, we actually have some liberty in how we implement such a derivation. E.g., the HTC-proof below applies TC rules below $\overline{TC}$ ones, and delays branching until the 'end' of proof search, which is impossible in TC_G. The only infinite branch, looping on $\bullet$, is progressing by the blue hypertrace.

$$\begin{prooftree}
\AxiomC{}
\LeftLabel{init}
\UnaryInfC{$\{\,\}^{\varnothing}$}
\LeftLabel{id}
\UnaryInfC{$\{a(c,d)\}^{\varnothing}, \{\bar{a}(c,d)\}^{\varnothing}$}
\AxiomC{}
\LeftLabel{init}
\UnaryInfC{$\{\,\}^{\varnothing}$}
\LeftLabel{id}
\UnaryInfC{$\{a(c,e)\}^{\varnothing}, \{\bar{a}(c,e)\}^{\varnothing}$}
\AxiomC{$\vdots$}
\LeftLabel{TC}
\RightLabel{$\bullet$}
\UnaryInfC{${\color{blue}\{\overline{TC}(a)(e,d)\}^{\varnothing}}, \{TC(\bar{a})(e,d)\}^{\varnothing}$}
\LeftLabel{$\cup$}
\BinaryInfC{$\{a(c,e)\}^{\varnothing}, {\color{blue}\{\overline{TC}(a)(e,d)\}^{\varnothing}}, \{\bar{a}(c,e), TC(\bar{a})(e,d)\}^{\varnothing}$}
\LeftLabel{$2\cup$}
\BinaryInfC{$\{a(c,d), a(c,e)\}^{\varnothing}, {\color{blue}\{a(c,d), \overline{TC}(a)(e,d)\}^{\varnothing}}, \{\bar{a}(c,d)\}^{\varnothing}, \{\bar{a}(c,e), TC(\bar{a})(e,d)\}^{\varnothing}$}
\LeftLabel{inst}
\UnaryInfC{$\{a(c,d), a(c,e)\}^{\varnothing}, {\color{blue}\{a(c,d), \overline{TC}(a)(e,d)\}^{\varnothing}}, \{\bar{a}(c,d)\}^{\varnothing}, \{\bar{a}(c,x), TC(\bar{a})(x,d)\}^{x}$}
\LeftLabel{$\overline{TC}$}
\UnaryInfC{${\color{blue}\{\overline{TC}(a)(c,d)\}^{\varnothing}}, \{\bar{a}(c,d)\}^{\varnothing}, \{\bar{a}(c,x), TC(\bar{a})(x,d)\}^{x}$}
\LeftLabel{TC}
\RightLabel{$\bullet$}
\UnaryInfC{${\color{blue}\{\overline{TC}(a)(c,d)\}^{\varnothing}}, \{TC(\bar{a})(c,d)\}^{\varnothing}$}
\end{prooftree}$$

This is an example of the more general 'rule permutations' available in HTC, hinting at a more flexible proof theory (we discuss this further in Sect. 8).

Example 21 (Transitivity). TC can be proved transitive by way of a cyclic proof in TC_G of the sequent $\overline{TC}(a)(c,d), \overline{TC}(a)(d,e), TC(\bar{a})(c,e)$. As in the previous example we may mimic that proof line by line, but we give a slightly different one that cannot directly be interpreted as a TC_G proof:

$$\begin{prooftree}
\AxiomC{}
\LeftLabel{init}
\UnaryInfC{$\{\,\}^{\varnothing}$}
\LeftLabel{id}
\UnaryInfC{$a(c,d), \bar{a}(c,d)$}
\AxiomC{Ex. 20}
\doubleLine
\UnaryInfC{$TC(\bar{a})(d,e), \overline{TC}(a)(d,e)$}
\LeftLabel{$\cup$}
\BinaryInfC{$a(c,d), \overline{TC}(a)(d,e), \bar{a}(c,d), TC(\bar{a})(d,e)$}
\LeftLabel{inst}
\UnaryInfC{$a(c,d), \overline{TC}(a)(d,e), \{\bar{a}(c,x), TC(\bar{a})(x,e)\}^{x}$}
\AxiomC{}
\LeftLabel{init}
\UnaryInfC{$\{\,\}^{\varnothing}$}
\LeftLabel{id}
\UnaryInfC{$a(c,c'), \bar{a}(c,c')$}
\AxiomC{$\vdots$}
\LeftLabel{TC}
\RightLabel{$\circ$}
\UnaryInfC{${\color{red}\overline{TC}(a)(c',d)}, \overline{TC}(a)(d,e), TC(\bar{a})(c',e)$}
\LeftLabel{$\cup$}
\BinaryInfC{$a(c,c'), {\color{red}\overline{TC}(a)(c',d)}, \overline{TC}(a)(d,e), \bar{a}(c,e), \bar{a}(c,c'), TC(\bar{a})(c',e)$}
\LeftLabel{inst}
\UnaryInfC{$a(c,c'), {\color{red}\overline{TC}(a)(c',d)}, \overline{TC}(a)(d,e), \bar{a}(c,e), \{\bar{a}(c,x), TC(\bar{a})(x,e)\}^{x}$}
\LeftLabel{$2\cup$}
\BinaryInfC{$a(c,d), a(c,c'), {\color{red}a(c,d), \overline{TC}(a)(c',d)}, \overline{TC}(a)(d,e), \bar{a}(c,e), \{\bar{a}(c,x), TC(\bar{a})(x,e)\}^{x}$}
\LeftLabel{$\overline{TC}$}
\UnaryInfC{${\color{red}\overline{TC}(a)(c,d)}, \overline{TC}(a)(d,e), \bar{a}(c,e), \{\bar{a}(c,x), TC(\bar{a})(x,e)\}^{x}$}
\LeftLabel{TC}
\RightLabel{$\circ$}
\UnaryInfC{${\color{red}\overline{TC}(a)(c,d)}, \overline{TC}(a)(d,e), TC(\bar{a})(c,e)$}
\end{prooftree}$$

The only infinite branch (except for that from Example 20), looping on $\circ$, is progressing by the red hypertrace.

Finally, it is pertinent to revisit the 'counterexample' (4) that witnessed incompleteness of TC_G for PDL^{+}. The following result is, in fact, already implied by our later completeness result, Theorem 28, but we shall present it nonetheless:

Proposition 22. $\mathsf{HTC} \vdash_{cyc} \mathsf{ST}((aa \cup aba)^{+})(c,d) \supset \mathsf{ST}(a^{+}((ba^{+})^{+} \cup a))(c,d)$.

Proof. We give the required cyclic proof in Fig. 2, using the abbreviations: $\alpha(c,d) = \mathsf{ST}(aa \cup aba)(c,d)$ and $\beta(c,d) = \mathsf{ST}((ba^+)^+ \cup a)(c,d)$. The only infinite branch (looping on $\bullet$) has progressing hypertrace is marked in blue.
Hypersequents $\mathbf{R} = \{\overline{\alpha}(c,d)\}^{\varnothing}, \{\overline{\alpha}(c,d), \overline{TC}(\overline{\alpha})(e,d)\}^{\varnothing}, \{TC(a)(c,y), \beta(y,d)\}^{y}$ and $\mathbf{R}' = \{\overline{\alpha}(c,d)\}^{\varnothing}, \{\overline{\alpha}(c,d)\}^{\varnothing}, \{TC(a)(c,y), \beta(y,d)\}^{y}$ have finitary proofs, while $\mathbf{P} = \{\overline{aba}(c,e)\}^{\varnothing}, \{\overline{TC}(\overline{\alpha})(e,d)\}^{\varnothing}, \{TC(a)(c,y), \beta(y,d)\}^{y}$ has a cyclic proof.

$$
\begin{prooftree}
\AxiomC{}
\RightLabel{$\bullet$}
\UnaryInfC{$\{\overline{TC}(\overline{\alpha})(e,d)\}^{\varnothing}, \{TC(a)(j,e), \beta(yd)\}^{y}$}
\RightLabel{id}
\UnaryInfC{$\{\overline{a}(f,e)\}^{\varnothing}, \{\overline{TC}(\overline{\alpha})(e,d)\}^{\varnothing}, \{a(f,g), TC(a)(g,y), \beta(y,d)\}^{y}$}
\RightLabel{inst}
\UnaryInfC{$\{\overline{a}(f,e)\}^{\varnothing}, \{\overline{TC}(\overline{\alpha})(e,d)\}^{\varnothing}, \{a(f,k), TC(a)(k,y), \beta(y,d)\}^{y,k}$}
\RightLabel{$[e/k]$}
\RightLabel{wk}
\UnaryInfC{$\{\overline{a}(f,e)\}^{\varnothing}, \{\overline{TC}(\overline{\alpha})(e,\ d)\}^{\varnothing}, \{a(f,y), \beta(y,d)\}^{y}, \{a(f,k), TC(a)(k,y), \beta(y,d)\}^{y,k}$}
\RightLabel{TC}
\UnaryInfC{$\{\overline{a}(f,e)\}^{\varnothing}, \{\overline{TC}(\overline{\alpha})(e,d)\}^{\varnothing}, \{TC(a)(f,y), \beta(y,d)\}^{y}$}
\RightLabel{id}
\UnaryInfC{$\{\overline{a}(c,f)\}^{\varnothing}, \{\overline{a}(f,e)\}^{\varnothing}, \{\overline{TC}(\overline{\alpha})(e,d)\}^{\varnothing}, \{a(c,f), TC(a)(f,y), \beta(y,d)\}^{y}$}
\RightLabel{inst}
\UnaryInfC{$\{\overline{a}(c,f)\}^{\varnothing}, \{\overline{a}(f,e)\}^{\varnothing}, \{\overline{TC}(\overline{\alpha})(e,d)\}^{\varnothing}, \{a(c,z), TC(a)(z,y), \beta(y,d)\}^{y,z}$}
\RightLabel{$[f/z]$}
\RightLabel{wk}
\UnaryInfC{$\{\overline{a}(c,f)\}^{\varnothing}, \{\overline{a}(f,e)\}^{\varnothing}, \{\overline{TC}(\overline{\alpha})(e,d)\}^{\varnothing}, \{a(c,y), \beta(y,d)\}^{y}, \{a(c,z), TC(a)(z,y), \beta(y,d)\}^{y,z}$}
\RightLabel{TC}
\UnaryInfC{$\{\overline{a}(c,f)\}^{\varnothing}, \{\overline{a}(f,e)\}^{\varnothing}, \{TC(a)(c,y), \beta(y,d)\}^{y}$}
\RightLabel{$\forall, \vee$}
\UnaryInfC{$\{\overline{aa}(c,e)\}^{\varnothing}, \{\overline{TC}(\overline{\alpha})(e,d)\}^{\varnothing}, \{TC(a)(c,y), \beta(y,d)\}^{y}$}
\RightLabel{$\cup$ $\mathbf{P}$}
\UnaryInfC{$\{\overline{aa}(c,e), \overline{aba}(c,e)\}^{\varnothing}, \{\overline{TC}(\overline{\alpha})(e,d)\}^{\varnothing}, \{TC(a)(c,y), \beta(y,d)\}^{y}$}
\RightLabel{$\wedge$}
\UnaryInfC{$\{\overline{\alpha}(c,e)\}^{\varnothing}, \{\overline{TC}(\overline{\alpha})(e,d)\}^{\varnothing}, \{TC(a)(c,y), \beta(y,d)\}^{y}$}
\RightLabel{$\cup$ $\mathbf{R}'$}
\UnaryInfC{$\{\overline{\alpha}(c,e)\}^{\varnothing}, \{\overline{\alpha}(c,d), \overline{TC}(\overline{\alpha})(e,d)\}^{\varnothing}, \{TC(a)(,y), \beta(y,d)\}^{y}$}
\RightLabel{$\cup$ $\mathbf{R}$}
\UnaryInfC{$\{\overline{\alpha}(c,d), \overline{\alpha}(c,e)\}^{\varnothing}, \{\overline{\alpha}(c,d), \overline{TC}(\overline{\alpha})(e,d)\}^{\varnothing}, \{TC(a)(c,y), \beta(y,d)\}^{y}$}
\RightLabel{TC}
\UnaryInfC{$\{\overline{TC}(\overline{\alpha})(c,d)\}^{\varnothing}, \{TC(a)(c,y), \beta(y,d)\}^{y}$}
\RightLabel{$\bullet$}
\RightLabel{$\vee, \exists, \wedge$}
\UnaryInfC{$\{\mathsf{ST}((aa \cup aba)^+)(c,d) \vee \mathsf{ST}(a^+; ((ba^+)^+ \cup a))(c,d)\}^{\varnothing}$}
\end{prooftree}
$$

Fig. 2. Cyclic proof for sequent not cyclically provable by TC_G.

5 Soundness of HTC

This section is devoted to the proof of the first of our main results:

Theorem 23 (Soundness). *If* $\mathsf{HTC} \vdash_{nwf} \mathbf{S}$ *then* $\models \mathbf{S}$.

The argument is quite technical due to the alternating nature of our progress condition. In particular the treatment of traces within hypertraces requires a more fine grained argument than usual, bespoke to our hypersequential structure.

Throughout this section, we shall fix a HTC preproof $\mathcal{D}$ of a hypersequent $\mathbf{S}$. For practical reasons we shall assume that $\mathcal{D}$ is substitution-free (at the cost of regularity) and that each quantifier in $\mathbf{S}$ binds a distinct variable.[4] We further assume some structure $\mathcal{M}^{\times}$ and an interpretation ρ_0 such that $\rho_0 \not\models \mathbf{S}$ (within $\mathcal{M}^{\times}$). Since each rule is locally sound, by contraposition we can continually choose 'false premisses' to construct an infinite 'false branch':

Lemma 24 (Countermodel branch). *There is a branch* $\mathcal{B}^{\times} = (\mathbf{S}_i)_{i<\omega}$ *of* $\mathcal{D}$ *and an interpretation* $\rho^{\times}$ *such that, with respect to* $\mathcal{M}^{\times}$*:*

[4] Note that this convention means we can simply take $y = x$ in the $\exists$ rule in Fig. 1.

1. *$\rho^\times \not\models \mathbf{S}_i$, for all $i < \omega$;*
2. *Suppose that $\mathbf{S}_i$ concludes a $\overline{TC}$ step, as typeset in Fig. 1, and $\rho^\times \models TC(\bar{A})(s,t)\,[\mathbf{d}/\mathbf{x}]$. If n is minimal such that $\rho^\times \models \bar{A}(d_i, d_{i+1})$ for all $i \leq n$, $\rho^\times(s) = d_0$ and $\rho^\times(t) = d_n$, and $n > 1$, then $\rho^\times(f)(\mathbf{d}) = d_1$[5] so that $\rho_{i+1} \models \bar{A}(s, f(\mathbf{x}))[\mathbf{d}/\mathbf{x}]$ and $\rho^\times \models TC(\bar{A})(f(\mathbf{x}), t)[\mathbf{d}/\mathbf{x}]$.*

Unpacking this a little, our interpretation $\rho^\times$ is actually defined as the limit of a chain of 'partial' interpretations $(\rho_i)_{i<\omega}$, with each $\rho_i \not\models \mathbf{S}_i$ (within $\mathcal{M}^\times$). Note in particular that, by 2, whenever some $\overline{TC}$-formula is principal, we choose ρ_{i+1} to always assign to it a falsifying path of minimal length (if one exists at all), with respect to the assignment to variables in its annotation. It is crucial at this point that our definition of $\rho^\times$ is parametrised by such assignments.

Let us now fix $\mathcal{B}^\times$ and $\rho^\times$ as provided by the Lemma above. Moreover, let us henceforth assume that $\mathcal{D}$ is a proof, i.e. it is progressing, and fix a progressing hypertrace $\mathcal{H} = (\{\Gamma_i\}^{\mathbf{x}_i})_{i<\omega}$ along $\mathcal{B}^\times$. In order to carry out an infinite descent argument, we will need to define a particular trace along this hypertrace that 'preserves' falsity, bottom-up. This is delicate since the truth values of formulas in a trace depend on the assignment of elements to variables in the annotations. A particular issue here is the instantiation rule inst, which requires us to 'revise' whatever assignment of y we may have defined until that point. Thankfully, our earlier convention on substitution-freeness and uniqueness of bound variables in $\mathcal{D}$ facilitates the convergence of this process to a canonical such assignment:

Definition 25 (Assignment). *We define $\delta_\mathcal{H} : \bigcup_{i<\omega} \mathbf{x}_i \to |\mathcal{M}^\times|$ by $\delta_\mathcal{H}(x) := \rho(t)$ if x is instantiated by t in $\mathcal{H}$; otherwise $\delta_\mathcal{H}(x)$ is some arbitrary $d \in |\mathcal{M}^\times|$.*

Note that $\delta_\mathcal{H}$ is indeed well-defined, thanks to the convention that each quantifier in $\mathbf{S}$ binds a distinct variable. In particular we have that each variable x is instantiated at most once along a hypertrace. Henceforth we shall simply write $\rho, \delta_\mathcal{H} \models A(\mathbf{x})$ instead of $\rho \models A(\delta_\mathcal{H}(\mathbf{x}))$. Working with such an assignment ensures that false formulas along $\mathcal{H}$ always have a false immediate ancestor:

Lemma 26 (Falsity through $\mathcal{H}$). *If $\rho^\times, \delta_\mathcal{H} \not\models F$ for some $F \in \Gamma_i$, then F has an immediate ancestor $F' \in \Gamma_{i+1}$ with $\rho^\times, \delta_\mathcal{H} \not\models F'$.*

In particular, regarding the inst rule of Fig. 1, note that if $F \in \Gamma(y)$ then we can choose $F' = F[t/y]$ which, by definition of $\delta_\mathcal{H}$, has the same truth value. By repeatedly applying this Lemma we obtain:

Proposition 27 (False trace). *There exists an infinite trace $\tau^\times = (F_i)_{i<\omega}$ through $\mathcal{H}$ such that, for all i, it holds that $\mathcal{M}^\times, \rho^\times, \delta_\mathcal{H} \not\models F_i$.*

We are now ready to prove our main soundness result.

Proof (of Theorem 23, sketch). Fix the infinite trace $\tau^\times = (F_i)_{i<\omega}$ through $\mathcal{H}$ obtained by Proposition 27. Since $\tau^\times$ is infinite, by definition of HTC proofs, it

[5] To be clear, we here choose an arbitrary such minimal '$\bar{A}$-path'.

needs to be progressing, i.e., it is infinitely often $\overline{TC}$-principal and there is some $k \in \mathbb{N}$ s.t. for $i > k$ we have that $F_i = \overline{TC}(A)(s_i, t_i)$ for some terms s_i, t_i.

To each F_i, for $i > k$, we associate the natural number n_i measuring the '$\bar{A}$-distance between s_i and t_i'. Formally, $n_i \in \mathbb{N}$ is least such that there are $d_0, \ldots, d_{n_i} \in |\mathcal{M}^\times|$ with $\rho^\times(s) = d_0, \rho^\times(t) = d_{n_i}$ and, for all $i < n_i$, $\rho^\times, \delta_{\mathcal{H}} \models \bar{A}(d_i, d_{i+1})$. Our aim is to show that $(n_i)_{i>k}$ has no minimal element, contradicting wellfoundness of $\mathbb{N}$. For this, we establish the following two local properties:

$$\mathsf{id}\,\frac{}{p, \bar{p}} \qquad \mathsf{wk}\,\frac{\Gamma}{\Gamma, A} \qquad \mathsf{k}_a\,\frac{\Gamma, A}{\langle a\rangle\Gamma, [a]A} \qquad \wedge\,\frac{\Gamma, A \quad \Gamma, B}{\Gamma, A \wedge B} \qquad \vee_0\,\frac{\Gamma, A_0}{\Gamma, A_0 \vee A_1} \qquad \vee_1\,\frac{\Gamma, A_1}{\Gamma, A_0 \vee A_1}$$

$$\langle;\rangle\,\frac{\Gamma, \langle\alpha\rangle\langle\beta\rangle A}{\Gamma, \langle\alpha;\beta\rangle A} \qquad \langle\cup\rangle_0\,\frac{\Gamma, \langle\alpha_0\rangle A}{\Gamma, \langle\alpha_0 \cup \alpha_1\rangle A} \qquad \langle\cup\rangle_1\,\frac{\Gamma, \langle\alpha_1\rangle A}{\Gamma, \langle\alpha_0 \cup \alpha_1\rangle A} \qquad [\cup]\,\frac{\Gamma, [\alpha]A \quad \Gamma, [\beta]A}{\Gamma, [\alpha \cup \beta]A}$$

$$[;]\,\frac{\Gamma, [\alpha][\beta]A}{\Gamma, [\alpha;\beta]A} \qquad \langle+\rangle_0\,\frac{\Gamma, \langle\alpha\rangle A}{\Gamma, \langle\alpha^+\rangle A} \qquad \langle+\rangle_1\,\frac{\Gamma, \langle\alpha\rangle\langle\alpha^+\rangle A}{\Gamma, \langle\alpha^+\rangle A} \qquad [+]\,\frac{\Gamma, [\alpha]A \quad \Gamma, [\alpha][\alpha^+]A}{\Gamma, [\alpha^+]A}$$

Fig. 3. Rules of LPD^+.

1. $(n_i)_{i>k}$ is *monotone decreasing*, i.e., for all $i > k$, we have $n_{i+1} \leq n_i$;
2. Whenever F_i is principal, we have $n_{i+1} < n_i$.

So $(n_i)_{i>k}$ is monotone decreasing, by 1, but cannot converge, by 2 and the definition of progressing trace. Thus $(n_i)_{k<i}$ has no minimal element, yielding the required contradiction.

6 HTC is Complete for PDL$^+$, Over Standard Translation

In this section we give our next main result:

Theorem 28 (Completeness for PDL$^+$). *For a* PDL$^+$ *formula* A*, if* $\models A$ *then* $\mathsf{HTC} \vdash_{cyc} \mathsf{ST}(A)(c)$.

The proof is by a direct simulation of a cut-free cyclic system for PDL$^+$ that is complete. We shall briefly sketch this system below.

6.1 Circular System for PDL$^+$

The system LPD^+, given in Fig. 3, is the natural extension of the usual sequent calculus for basic multimodal logic K by rules for programs. In Fig. 3, $\langle a\rangle\Gamma$ is shorthand for $\{\langle a\rangle B : B \in \Gamma\}$. (Regular) preproofs for this system are defined just like for HTC or TC_G. The notion of 'immediate ancestor' is induced by the indicated colouring: a formula C in a premiss is an immediate ancestor of a formula C' in the conclusion if they have the same colour; if $C, C' \in \Gamma$ then we furthermore require $C = C'$.

Definition 29 (Non-wellfounded proofs). *Fix a preproof $\mathcal{D}$ of a sequent Γ. A* thread *is a maximal path in its graph of immediate ancestry. We say a thread is* progressing *if it has a smallest infinitely often principal formula of the form $[\alpha^+]A$. $\mathcal{D}$ is a* proof *if every infinite branch has a progressing thread. If $\mathcal{D}$ is regular, we call it a* cyclic proof *and we may write $\mathsf{LPD}^+ \vdash_{cyc} \Gamma$.*

Soundness of cyclic-LPD^+ is established by a standard infinite descent argument, but is also implied by the soundness of cyclic-HTC (Theorem 23) and the simulation we are about to give (Theorem 28), though this is somewhat overkill. Completeness may be established by the game theoretic approach of Niwinskí and Walukiewicz [23], as done by Lange [20] for PDL (with identity), or by purely proof theoretic techniques of Studer [25]. Either way, both results follow from a standard embedding of PDL$^+$ into the μ-calculus and its known completeness results [23,25], by way of a standard 'proof reflection' argument: μ-calculus proofs of the embedding are 'just' step-wise embeddings of LPD^+ proofs:

Theorem 30 (Soundness and completeness, [20]). *Let A be a* PDL$^+$ *formula. $\models A$ iff $\mathsf{LPD}^+ \vdash_{cyc} A$.*

6.2 A 'Local' Simulation of LPD^+ by HTC

In this subsection we show that LPD^+-preproofs can be stepwise transformed into HTC-proofs, with respect to the standard translation. In order to produce this local simulation, we need a more refined version of the standard translation that incorporates the structural elements of hypersequents.

Fix a PDL$^+$ formula $A = [\alpha_1]\dots[\alpha_n]\langle\beta_1\rangle\dots\langle\beta_m\rangle B$, for $n, m \geq 0$. The *hypersequent translation* of A, written $\mathsf{HT}(A)(c)$, is defined as:

$$\{\overline{\mathsf{ST}(\alpha_1)(c,d_1)}\}^{\varnothing}, \{\overline{\mathsf{ST}(\alpha_2)(d_1,d_2)}\}^{\varnothing}, \dots, \{\overline{\mathsf{ST}(\alpha_n)(d_{n-1},d_n)}\}^{\varnothing},$$
$$\{\mathsf{ST}(\beta_1)(d_n,y_1), \mathsf{ST}(\beta_2)(y_2,y_3), \dots, \mathsf{ST}(\beta_m)(y_{m-1},y_m), \mathsf{ST}(B)(y_m)\}^{y_1,\dots,y_m}$$

For $\Gamma = A_1, \dots, A_k$, we write $\mathsf{HT}(\Gamma)(c) \coloneqq \mathsf{HT}(A_1)(c), \dots, \mathsf{HT}(A_k)(c)$.

Definition 31 (HT-translation). *Let $\mathcal{D}$ be a* PDL$^+$ *preproof. We shall define a HTC preproof $\mathsf{HT}(\mathcal{D})(c)$ of the hypersequent $\mathsf{HT}(A)(c)$ by a local translation of inference steps. We give only a few of the important cases here, but a full definition can be found in [13].*

– *A step* $\mathsf{k}_a \dfrac{B_1, \dots, B_k, A}{\langle a\rangle B_1, \dots, \langle a\rangle B_k, [a]A}$ *is translated to:*

$$\mathsf{inst}\,\mathsf{=}\,\dfrac{\cup\,\dfrac{\mathsf{wk}\,\dfrac{\vee,\forall\,\dfrac{[d/c]\,\dfrac{\mathsf{HT}(B_1)(c),\dots,\mathsf{HT}(B_k)(c),\mathsf{HT}(A)(c)}{\mathsf{HT}(B_1)(d),\dots,\mathsf{HT}(B_k)(d),\mathsf{HT}(A)(d)}}{\{\mathsf{CT}(B_1)(d)\}^{\mathbf{x}_{B_1}},\dots,\{\mathsf{CT}(B_k)(d)\}^{\mathbf{x}_{B_k}},\mathsf{HT}(A)(d)}}{\{\mathsf{CT}(B_1)(d)\}^{\mathbf{x}_{B_1}},\dots,\{\mathsf{CT}(B_k)(d)\}^{\mathbf{x}_{B_k}},\{\overline{\mathsf{ST}(a)(c,d)}\}^{\varnothing},\mathsf{HT}(A)(d)}}{\{\mathsf{CT}(B_1)(d)\}^{\mathbf{x}_{B_1}},\dots,\{\mathsf{ST}(a)(c,d),\mathsf{CT}(B_k)(d)\}^{\mathbf{x}_{B_k}},\{\overline{\mathsf{ST}(a)(c,d)}\}^{\varnothing},\mathsf{HT}(A)(d)}}{\dfrac{\{\mathsf{ST}(a)(c,y),\mathsf{CT}(B_1)(y)\}^{\mathbf{x}_{B_1},y},\dots,\{\mathsf{ST}(a)(c,y),\mathsf{CT}(B_k)(y)\}^{\mathbf{x}_{B_k},y},\{\overline{\mathsf{ST}(a)(c,d)}\}^{\varnothing},\mathsf{HT}(A)(d)}{\mathsf{HT}(\langle a\rangle B_1)(c),\dots,\mathsf{HT}(\langle a\rangle B_k)(c),\mathsf{HT}([a]A)(c)}}$$

where (omitted) left-premisses of $\cup$ *steps are simply proved by* $\mathsf{wk}, \mathsf{id}, \mathsf{init}$. *In this and the following cases, we use the notation* $\mathsf{CT}(A)(c)$ *and* $\mathbf{x}_A$ *for the appropriate sets of formulas and variables forced by the definition of* HT *(again, see [13] for further details).*

– *A* $\langle\cup\rangle_i$ *step (for* $i = 0, 1$*), as typeset in Fig. 3, is translated to:*

$$\dfrac{\dfrac{\mathsf{HT}(\Gamma)(c), \mathsf{HT}(\langle\alpha_i\rangle A)(c)}{\mathsf{HT}(\Gamma)(c), \{\mathsf{ST}(\alpha_i)(c,y), \mathsf{CT}(A)(y)\}^{\mathbf{x}_B, y}}\,=}{\mathsf{HT}(\Gamma)(c), \{\mathsf{ST}(\alpha_0)(c,y) \vee \mathsf{ST}(\alpha_1)(c,y), \mathsf{CT}(A)(y)\}^{\mathbf{x}_A, y}}\,\vee\,=$$

– *A* $\langle;\rangle$ *step, as typeset in Fig. 3, is translated to:*

$$\dfrac{\dfrac{\dfrac{\mathsf{HT}(\Gamma)(c), \mathsf{HT}(\langle\alpha\rangle\langle\beta\rangle A)(c)}{\mathsf{HT}(\Gamma)(c), \{\mathsf{ST}(\alpha)(c,z), \mathsf{ST}(\alpha)(z,y), \mathsf{CT}(A)(y)\}^{\mathbf{x}_A, y, z}}\,=}{\mathsf{HT}(\Gamma)(c), \{\mathsf{ST}(\alpha)(c,z) \wedge \mathsf{ST}(\alpha)(z,y), \mathsf{CT}(A)(y)\}^{\mathbf{x}_A, y, z}}\,\wedge}{\mathsf{HT}(\Gamma)(c), \{\exists z(\mathsf{ST}(\alpha)(c,z) \wedge \mathsf{ST}(\alpha)(z,y)), \mathsf{CT}(A)(y)\}^{\mathbf{x}_A, y}}\,\exists\,=$$

– *A* $[+]$ *step, as typeset in Fig. 3, is translated to:*

$$\dfrac{\dfrac{\dfrac{\mathcal{E}\quad \dfrac{\mathcal{E}'\quad \dfrac{\mathsf{HT}(\Gamma)(c), \mathsf{HT}([\alpha][\alpha^+]A)(c)}{\mathsf{HT}(\Gamma)(c), \{\overline{\mathsf{ST}(\alpha)(c,f)}\}^{\varnothing}, \{\overline{TC}(\overline{\mathsf{ST}(\alpha)})(f,d)\}^{\varnothing}, \mathsf{HT}(A)(d)}\,=}{\mathsf{HT}(\Gamma)(c), \{\overline{\mathsf{ST}(\alpha)(c,f)}\}^{\varnothing}, \{\overline{\mathsf{ST}(\alpha)(c,d)}, \overline{TC}(\overline{\mathsf{ST}(\alpha)})(f,d)\}^{\varnothing}, \mathsf{HT}(A)(d)}\,\cup}{\mathsf{HT}(\Gamma)(c), \{\overline{\mathsf{ST}(\alpha)(c,d)}, \overline{\mathsf{ST}(\alpha)(c,f)}\}^{\varnothing}, \{\overline{\mathsf{ST}(\alpha)(c,d)}, \overline{TC}(\overline{\mathsf{ST}(\alpha)})(f,d)\}^{\varnothing}, \mathsf{HT}(A)(d)}\,\cup}{\mathsf{HT}(\Gamma)(c), \{\overline{TC}(\overline{\mathsf{ST}(\alpha)})(c,d)\}^{\varnothing}, \mathsf{HT}(A)(d)}\,\overline{TC}}{\mathsf{HT}(\Gamma)(c), \mathsf{HT}([\alpha^+]A)(c)}\,=$$

where $\mathcal{E}$ *and* $\mathcal{E}'$ *derive* $\mathsf{HT}(\Gamma)(c)$ *and* $\mathsf{HT}([\alpha]A)(c)$*, resp., using* wk*-steps.*

Note that, formally speaking, the well-definedness of $\mathsf{HT}(\mathcal{D})(c)$ in the definition above is guaranteed by coinduction: each rule of $\mathcal{D}$ is translated into a (nonempty) derivation.

Remark 32 (Deeper inference). Observe that HTC can also simulate 'deeper' program rules than are available in LPD^+. E.g. a rule $\dfrac{\Gamma, \langle\alpha\rangle\langle\beta_i\rangle A}{\Gamma, \langle\alpha\rangle\langle\beta_0 \cup \beta_1\rangle A}$ may be simulated too (similarly for $[\,]$). E.g. $\langle a^+\rangle\langle b\rangle p \supset \langle a^+\rangle\langle b \cup c\rangle p$ admits a *finite* proof in HTC (under ST), rather than a necessarily infinite (but cyclic) one in LPD^+.

6.3 Justifying Regularity and Progress

Proposition 33. *If* $\mathcal{D}$ *is regular, then so is* $\mathsf{HT}(\mathcal{D})(c)$.

Proof. Notice that each rule in $\mathcal{D}$ is translated to a finite derivation in $\mathsf{HT}(\mathcal{D})(c)$. Thus, if $\mathcal{D}$ has only finitely many distinct subproofs, then also $\mathsf{HT}(\mathcal{D})(c)$ has only finitely many distinct subproofs.

Proposition 34. *If $\mathcal{D}$ is progressing, then so is* $\mathsf{HT}(\mathcal{D})(c)$.

Proof (sketch). We need to show that every infinite branch of $\mathsf{HT}(\mathcal{D})(c)$ has a progressing hypertrace. Since the HT translation is defined stepwise on the individual steps of $\mathcal{D}$, we can associate to each infinite branch $\mathcal{B}$ of $\mathsf{HT}(\mathcal{D})(c)$ a unique infinite branch $\mathcal{B}'$ of $\mathcal{D}$. Since $\mathcal{D}$ is progressing, let $\tau = (F_i)_{i<\omega}$ be a progressing thread along $\mathcal{B}'$. By inspecting the rules of LPD^+ (and by definition of progressing thread), for some $k \in \mathbb{N}$, each F_i for $i > k$ has the form: $[\alpha_{i,1}] \cdots [\alpha_{i,n_i}][\alpha^+]A$, for some $n_i \geq 0$. So, for $i > k$, $\mathsf{HT}(F_i)(d_i)$ has the form:

$$\{\overline{\mathsf{ST}(\alpha_{i,1})(c, d_{i,1})}\}^{\varnothing}, \ldots, \{\overline{\mathsf{ST}(\alpha_{i,n_i})(d_{i,n_i-1}, d_{i,n_i})}\}^{\varnothing}, \{\overline{TC}(\overline{\mathsf{ST}(\alpha)})(d_{i,n_i}, d_i)\}^{\varnothing}, \mathsf{HT}(A)(d_i)$$

By inspection of the HT-translation (Definition 31) whenever F_{i+1} is an immediate ancestor of F_i in $\mathcal{B}'$, there is a path from the cedent $\{\overline{TC}(\overline{\mathsf{ST}(\alpha)})(d_{i+1,n_{i+1}}, d_{i+1})\}^{\varnothing}$ to the cedent $\{\overline{TC}(\overline{\mathsf{ST}(\alpha)})(d_{i,n_i}, d_i)\}^{\varnothing}$ in the graph of immediate ancestry along $\mathcal{B}$. Thus, since $\tau = (F_i)_{i<\omega}$ is a trace along $\mathcal{B}'$, we have a (infinite) hypertrace of the form $\mathcal{H}_\tau := (\{\Delta_i, \overline{TC}(\overline{\mathsf{ST}(\alpha)})(d_{i,n_i}, d_i)\}^{\varnothing})_{i>k'}$ along $\mathcal{B}$. By construction $\Delta_i = \varnothing$ for infinitely many $i > k'$, and so $\mathcal{H}_\tau$ has just one infinite trace. Moreover, by inspection of the $[+]$ step in Definition 31, this trace progresses in $\mathcal{B}$ every time τ does in $\mathcal{B}'$, and so progresses infinitely often. Thus, $\mathcal{H}$ is a progressing hypertrace. Since the choice of the branch $\mathcal{B}$ of $\mathcal{D}$ was arbitrary, we are done.

6.4 Putting it all Together

We can now finally conclude our main simulation theorem:

Proof (of Theorem 28, sketch). Let A be a PDL^+ formula s.t. $\models A$. By the completeness result for LPD^+, Theorem 30, we have that $\mathsf{LPD}^+ \vdash_{cyc} A$, say by a cyclic proof $\mathcal{D}$. From here we construct the HTC preproof $\mathsf{HT}(\mathcal{D})(c)$ which, by Propositions 33 and 34, is in fact a cyclic proof of $\mathsf{HT}(A)(c)$. Finally, we apply some basic $\vee, \wedge, \exists, \forall$ steps to obtain a cyclic HTC proof of $\mathsf{ST}(A)(c)$.

7 Extension by Equality and Simulating Full PDL

We now briefly explain how our main results are extended to the 'reflexive' version of TCL. The language of $\mathsf{HTC}_=$ allows further atomic formulas of the form $s = t$ and $s \neq t$. The calculus $\mathsf{HTC}_=$ extends HTC by the rules:

$$=\frac{\mathbf{S}, \{\Gamma\}^{\mathsf{x}}}{\mathbf{S}, \{t = t, \Gamma\}^{\mathsf{x}}} \qquad \neq\frac{\mathbf{S}, \{\Gamma(s), \Delta(s)\}^{\mathsf{x}}}{\mathbf{S}, \{\Gamma(s), s \neq t\}^{\mathsf{x}}, \{\Delta(t)\}^{\mathsf{x}}}$$

The notion of immediate ancestry is colour-coded as in Definition 15, and the resulting notions of (pre)proof, (hyper)trace and progress are as in Definition 17. The simulation of Cohen and Rowe's system TC_G extends to

their reflexive system, RTC_G, by defining their operator $RTC(\lambda x, y.A)(s,t) \coloneqq TC(\lambda x, y.(x = y \vee A))(s,t)$. Note that, while it is semantically correct to set $RTC(A)(s,t)$ to be $s = t \vee TC(A)(s,t)$, this encoding does not lift to the Cohen-Rowe rules for RTC. Understanding that structures interpret $=$ as true equality, a modular adaptation of the soundness argument for HTC, cf. Sect. 5, yields:

Theorem 35 (Soundness of $\mathsf{HTC}_=$). *If* $\mathsf{HTC}_= \vdash_{nwf} \mathbf{S}$ *then* $\models \mathbf{S}$.

Turning to the modal setting, PDL may be defined as the extension of PDL^+ by including a program $A?$ for each formula A. Semantically, we have $(A?)^{\mathcal{M}} = \{(v,v) : \mathcal{M}, v \models A\}$. From here we may define $\varepsilon \coloneqq \top?$ and $\alpha^* \coloneqq (\varepsilon \cup \alpha)^+$; again, while it is semantically correct to set $\alpha^* = \varepsilon \cup \alpha^+$, this encoding does not lift to the standard sequent rules for $*$. The system LPD is obtained from LPD^+ by including the rules:

$$\langle ? \rangle \frac{\Gamma, A \quad \Gamma, B}{\Gamma, \langle A? \rangle B} \qquad [?] \frac{\Gamma, \bar{A}, B}{\Gamma, [A?]B}$$

Again, the notion of immediate ancestry is colour-coded as for LPD^+; the resulting notions of (pre)proof, thread and progress are as in Definition 29. Just like for LPD^+, a standard encoding of LPD into the μ-calculus yields its soundness and completeness, thanks to known sequent systems for the latter, cf. [23,25], but has also been established independently [20]. Again, a modular adaptation of the simulation of LPD^+ by HTC, cf. Sect. 6, yields:

Theorem 36 (Completeness for PDL). *Let A be a* PDL *formula. If* $\models A$ *then* $\mathsf{HTC}_= \vdash_{cyc} \mathsf{ST}(A)(c)$.

8 Conclusions

In this work we proposed a novel cyclic system HTC for Transitive Closure Logic (TCL) based on a form of hypersequents. We showed a soundness theorem for standard semantics, requiring an argument bespoke to our hypersequents. Our system is cut-free, rendering it suitable for automated reasoning via proof search. We showcased its expressivity by demonstrating completeness for PDL, over the standard translation. In particular, we demonstrated formally that such expressivity is not available in the previously proposed system TC_G of Cohen and Rowe (Theorem 12). Our system HTC locally simulates TC_G too (Theorem 19).

As far as we know, HTC is the first cyclic system employing a form of *deep inference* resembling *alternation* in automata theory, e.g. wrt. proof checking, cf. Proposition 18. It would be interesting to investigate the structural proof theory that emerges from our notion of hypersequent. As hinted at in Examples 20 and 21, our hypersequential system exhibits more liberal rule permutations than usual sequents, so we expect their *focussing* and *cut-elimination* behaviours to similarly be richer, cf. [21,22]. Note however that such investigations are rather pertinent for pure predicate logic (without TC): focussing and cut-elimination arguments do not typically preserve regularity of non-wellfounded proofs, cf. [2].

Finally, our work bridges the cyclic proof theories of (identity-free) PDL and (reflexive) TCL. With increasing interest in both modal and predicate cyclic proof theory, it would be interesting to further develop such correspondences.

Acknowledgements. The authors would like to thank Sonia Marin, Jan Rooduijn and Reuben Rowe for helpful discussions on matters surrounding this work.

References

1. Avron, A.: Transitive closure and the mechanization of mathematics. In: Kamareddine, F.D. (eds) Thirty Five Years of Automating Mathematics. Applied Logic Series, vol. 28, pp. 149–171. Springer, Dordrecht (2003). https://doi.org/10.1007/978-94-017-0253-9_7
2. Baelde, D., Doumane, A., Saurin, A.: Infinitary proof theory: the multiplicative additive case. In: Talbot, J., Regnier, L. (eds.) 25th EACSL Annual Conference on Computer Science Logic, CSL 2016, 29 August–1 September 2016, Marseille, France. LIPIcs, vol. 62, pp. 42:1–42:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2016). https://doi.org/10.4230/LIPIcs.CSL.2016.42
3. Berardi, S., Tatsuta, M.: Classical system of martin-lof's inductive definitions is not equivalent to cyclic proofs. CoRR abs/1712.09603 (2017). http://arxiv.org/abs/1712.09603
4. Blackburn, P., van Benthem, J.: Modal logic: a semantic perspective. In: Blackburn, P., van Benthem, J.F.A.K., Wolter, F. (eds.) Handbook of Modal Logic, Studies in Logic and Practical Reasoning, vol. 3, pp. 1–84. North-Holland (2007). https://doi.org/10.1016/s1570-2464(07)80004-8
5. Blackburn, P., De Rijke, M., Venema, Y.: Modal Logic, vol. 53. Cambridge University Press (2002)
6. Brotherston, J., Distefano, D., Petersen, R.L.: Automated cyclic entailment proofs in separation logic. In: Bjørner, N., Sofronie-Stokkermans, V. (eds.) CADE 2011. LNCS (LNAI), vol. 6803, pp. 131–146. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22438-6_12
7. Brotherston, J., Gorogiannis, N., Petersen, R.L.: A generic cyclic theorem prover. In: Jhala, R., Igarashi, A. (eds.) APLAS 2012. LNCS, vol. 7705, pp. 350–367. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-35182-2_25
8. Brotherston, J., Simpson, A.: Sequent calculi for induction and infinite descent. J. Log. Comput. **21**(6), 1177–1216 (2011)
9. Cohen, L., Rowe, R.N.S.: Uniform inductive reasoning in transitive closure logic via infinite descent. In: Ghica, D.R., Jung, A. (eds.) 27th EACSL Annual Conference on Computer Science Logic, CSL 2018, 4–7 September 2018, Birmingham, UK. LIPIcs, vol. 119, pp. 17:1–17:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2018). https://doi.org/10.4230/LIPIcs.CSL.2018.17
10. Cohen, L., Rowe, R.N.S.: Integrating induction and coinduction via closure operators and proof cycles. In: Peltier, N., Sofronie-Stokkermans, V. (eds.) IJCAR 2020. LNCS (LNAI), vol. 12166, pp. 375–394. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-51074-9_21
11. Cohen, L., Rowe, R.N.: Non-well-founded proof theory of transitive closure logic. ACM Trans. Comput. Log. **21**(4), 1–31 (2020)
12. Das, A.: On the logical complexity of cyclic arithmetic. Log. Methods Comput. Sci. **16**(1) (2020). https://doi.org/10.23638/LMCS-16(1:1)2020

13. Das, A., Girlando, M.: Cyclic proofs, hypersequents, and transitive closure logic (2022). https://doi.org/10.48550/ARXIV.2205.08616
14. Das, A., Pous, D.: A cut-free cyclic proof system for Kleene algebra. In: Schmidt, R.A., Nalon, C. (eds.) TABLEAUX 2017. LNCS (LNAI), vol. 10501, pp. 261–277. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66902-1_16
15. Grädel, E.: On transitive closure logic. In: Börger, E., Jäger, G., Kleine Büning, H., Richter, M.M. (eds.) CSL 1991. LNCS, vol. 626, pp. 149–163. Springer, Heidelberg (1992). https://doi.org/10.1007/BFb0023764
16. Gurevich, Y.: Logic and the Challenge of Computer Science, pp. 1–57. Computer Science Press (1988). https://www.microsoft.com/en-us/research/publication/logic-challenge-computer-science/
17. Immerman, N.: Languages that capture complexity classes. SIAM J. Comput. **16**(4), 760–778 (1987). https://doi.org/10.1137/0216051
18. Kozen, D.: A completeness theorem for Kleene algebras and the algebra of regular events. In: Proceedings of the Sixth Annual Symposium on Logic in Computer Science (LICS 1991), Amsterdam, The Netherlands, 15–18 July 1991, pp. 214–225. IEEE Computer Society (1991). https://doi.org/10.1109/LICS.1991.151646
19. Krob, D.: Complete systems of b-rational identities. Theor. Comput. Sci. **89**(2), 207–343 (1991). https://doi.org/10.1016/0304-3975(91)90395-I
20. Lange, M.: Games for modal and temporal logics. Ph.D. thesis (2003)
21. Marin, S., Miller, D., Volpe, M.: A focused framework for emulating modal proof systems. In: Beklemishev, L.D., Demri, S., Maté, A. (eds.) Advances in Modal Logic 11, Proceedings of the 11th Conference on "Advances in Modal Logic," held in Budapest, Hungary, 30 August–2 September 2016, pp. 469–488. College Publications (2016). http://www.aiml.net/volumes/volume11/Marin-Miller-Volpe.pdf
22. Miller, D., Volpe, M.: Focused labeled proof systems for modal logic. In: Davis, M., Fehnker, A., McIver, A., Voronkov, A. (eds.) LPAR 2015. LNCS, vol. 9450, pp. 266–280. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48899-7_19
23. Niwiński, D., Walukiewicz, I.: Games for the mu-calculus. Theor. Comput. Sci. **163**(1), 99–116 (1996). https://doi.org/10.1016/0304-3975(95)00136-0
24. Rowe, R.N.S., Brotherston, J.: Automatic cyclic termination proofs for recursive procedures in separation logic. In: Bertot, Y., Vafeiadis, V. (eds.) Proceedings of the 6th ACM SIGPLAN Conference on Certified Programs and Proofs, CPP 2017, Paris, France, 16–17 January 2017, pp. 53–65. ACM (2017). https://doi.org/10.1145/3018610.3018623
25. Studer, T.: On the proof theory of the modal mu-calculus. Stud. Logica. **89**(3), 343–363 (2008)
26. Tellez, G., Brotherston, J.: Automatically verifying temporal properties of pointer programs with cyclic proof. In: de Moura, L. (ed.) CADE 2017. LNCS (LNAI), vol. 10395, pp. 491–508. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63046-5_30
27. Tellez, G., Brotherston, J.: Automatically verifying temporal properties of pointer programs with cyclic proof. J. Autom. Reason. **64**(3), 555–578 (2020). https://doi.org/10.1007/s10817-019-09532-0