Labelled Process Logic

YUANRUI ZHANG, College of Software, Nanjing University of Aeronautics and Astronautics, China

This paper develops a cyclic labelled proof-theoretic framework for process logic — an extension of dynamic logic in which formulas specify properties of execution traces rather than only final states. The main difficulty is that first-order process logic must reason about concrete computations while preserving temporal information along regular-program traces. Existing compositional calculi cover important fragments, but do not provide a complete treatment of full first-order process logic over regular programs. We address this difficulty by enriching process-logic formulas with labels that explicitly record trace and update information during derivations. Based on this construction, we define cyclic labelled proof systems for propositional and first-order process logic, respectively denoted by G3PPL and G3FOPL. We prove the soundness by using the cyclic conditions to obtain an infinite descent in a well-founded multiset ordering, and prove the completeness by showing that the labelled systems can derive the established proof rules of process logic and first-order dynamic logic. The result is a uniform framework for process logic in which for the first time, trace-based program properties and first-order computations can be handled within the same proof structure.

CCS Concepts: • **Theory of computation** → **Modal and temporal logics**; **Programming logic**; **Logic and verification**; *Proof theory*.

Additional Key Words and Phrases: Process Logic, Program Logic, Temporal Properties, Labelled Proof Systems, Cyclic Proofs, Program Verification

ACM Reference Format:

Yuanrui Zhang. 2018. Labelled Process Logic. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX)*. ACM, New York, NY, USA, 51 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Process logic [11] extends dynamic logic [12] from reasoning about the final states of program executions to reasoning about the whole execution process. In ordinary dynamic logic, a formula $[\alpha]\phi$ states that every terminating execution of program α ends in a state satisfying ϕ. This view has supported a wide range of program logics, including logics for process algebra [14], programming languages [12], synchronous systems [5], hybrid systems [18], and many related verification frameworks. Process logic strengthens this setting by allowing ϕ to describe trace properties using linear temporal logic (LTL [19]): a formula $[\alpha]\phi$ states that every execution trace of α satisfies the temporal property ϕ. ϕ can speak about what happens during the execution rather than only after termination [11]. This makes process logic a natural formalism for specifications in which intermediate behavior is essential.

The propositional theory of process logic (PPL) has been thoroughly studied, including decidability and completeness [11]. The first-ordered level, however, is more delicate. First-order process logic (FOPL) interprets atomic programs as concrete computations, such as assignments over an arithmetic structure, and interprets atomic propositions as

Author's Contact Information: Yuanrui Zhang, yuanruizhang@nuaa.edu.cn, College of Software, Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu, China.

Manuscript submitted to ACM

predicates over program states. This expressiveness is necessary for reasoning about actual computational systems, but it also exposes a structural difficulty: the proof system of PPL does not lift smoothly to the first-order case (cf. [11]). In particular, for a sequential program $\alpha;\beta$, a proof of $[\alpha;\beta]\phi$ would require a compositional way to split the trace property ϕ across the executions of α and β. Such a split depends heavily on the shape of the LTL formula involving operator **n** (next), $\Box$ (always), $\Diamond$ (eventually), or U (until).

Several lines of work address fragments of this difficulty. Beckert's sequent calculus for first-order dynamic logic (FODL) with trace modalities studies formulas of the form $[|\alpha|]\phi$, which correspond to the special process-logic case $[\alpha]\Box\phi$ [4]. Its proof system is complete for deterministic while programs. The crucial compositional rule for sequence programs reduces $[\alpha;\beta]\Box\phi$ to obligations about the trace of α and the subsequent behavior of β, but this rule is specific to the always modality $\Box$. Later, [17] employs the trace modality of DLT in a differential temporal dynamic logic (to avoid ambiguity, we name it DTL_d) for the more general regular programs, but it remains open whether the proof system of DLT is complete for this broader class. [22] points out that the proof system of DLT is insufficient to derive every valid temporal properties of general sequential regular programs, and thus needs to be improved. A variant of DTL_d which we name DTL_d^2 [13] extends DLT to a richer collection of temporal modalities for reasoning about properties such as $[\alpha]\Diamond\phi$. Moreover, [1] establishes the completeness of DTL_d for the deterministic fragment of regular programs. However, ${\mathrm{DTL}_d}^2$ does not handle temporal operators such as until (U). Based on DLT, dynamic trace logic (DTL) [2] is the first to propose a complete calculus for all LTL operators, but only for deterministic while programs rather than the nondeterministic regular programs.

Consequently, none of these systems above provides a complete proof-theoretic treatment of full first-order process logic over regular programs.

This paper takes a different route. Instead of searching for increasingly complex compositional rules for temporal formulas, we enrich process-logic formulas with *labels* that explicitly record the relevant execution information. A labelled formula has the form $\sigma : \phi$, where σ represents a trace-like program configuration and ϕ is a process-logic formula. Labels make intermediate execution information available inside the proof system, so that the proof no longer needs to split every sequential trace by a special-purpose temporal rule at the point where the sequence is encountered. The labelled form is conservative over ordinary process logic: an unlabelled formula can be recovered by using a free trace variable as its label.

Based on this idea, we develop labelled proof systems for both propositional and first-order process logic. The propositional system, G3PPL, extends labelled sequent techniques for modal logics [15] and for non-wellfounded labelled proof systems for dynamic logic [9], but adapts them to the trace semantics of process logic. Since regular programs include iteration, proof search may generate cyclic derivations. We therefore equip the labelled systems with cyclic proof conditions in the style of cyclic proof theory [6, 7, 20]. The cyclic condition provides the mechanism that distinguishes sound cyclic reasoning from arbitrary finite graphs with unsound back-links.

To our best knowledge, our work is the first to provide a complete logical framework for first-order process logic over regular programs.

The contributions of the paper are as follows.

- We introduce labelled versions of process-logic formulas, defining trace labels for PPL and update-based labels for FOPL.
- We define cyclic labelled proof systems G3PPL and G3FOPL, sharing a common labelled framework while using rules specific to propositional and first-order process logic.

- We prove the soundness for the cyclic labelled systems by reducing every progressive infinite derivation path to an infinite descent in a well-founded multiset ordering.
- We prove the completeness of G3PPL by deriving the proof system of PPL, and establish the corresponding first-order result for G3FOPL through the usual arithmetical encoding of FODL.

The rest of this paper is organized as follows. Section 2 first recalls the syntax, semantics, and proof-theoretic background of process logic. Section 3 then provides an overview: it motivates our labelled approach by showing how classical and existing trace-based program logics are embed into process logic and what are their limitations in details, and describes the high-level design of our work. Section 4 introduces labelled PPL formulas and the proof system G3PPL, followed by the analysis and proofs of its soundness and completeness in Section 5 and 6 respectively. The first-order labelled system G3FOPL is developed afterwards in Section 7, where we also analyze its soundness and completeness. The appendix contains the detailed derivations and auxiliary proofs used by the main theorems.

2 Brief Introduction to Process Logic

This section recalls the parts of process logic that are used in the rest of the paper. We first review propositional process logic [11], including its syntax, trace semantics, and proof system, and then describe the first-order instance over arithmetic that is used when defining labelled FOPL formulas.

2.1 Propositional Process Logic

Definition 2.1 (Syntax of PPL Formula). *Let a range over atomic programs and p range over atomic propositions. The syntax of programs, trace formulas, and state formulas is given by:*

$$\alpha =_{df} a \mid \phi? \mid \alpha\,;\alpha \mid \alpha \cup \alpha \mid \alpha^*,$$
$$\pi =_{df} \phi \mid \mathbf{f}\,\pi \mid \pi\ \mathbf{suf}\ \pi,$$
$$\phi =_{df} \mathit{true} \mid p \mid \neg\phi \mid \phi \wedge \phi \mid [\alpha]\pi,$$

Here a is an atomic program, also called an *action*. The program $\phi?$ is a test, $\alpha\,;\beta$ is sequential composition, $\alpha \cup \beta$ is nondeterministic choice, and α^* is finite iteration. The modality $[\alpha]\pi$ states that every trace generated by executing α satisfies the trace formula π.

The symbol p denotes an atomic proposition. A trace formula π is evaluated over a finite trace rather than only over a single state. The formula $\mathbf{f}\,\pi$ states that π holds at the first state of the current trace, while $\pi_1\ \mathbf{suf}\ \pi_2$ is the process-logic until-like suffix connective (cf. [11]): along a proper suffix of the current trace, π_2 eventually holds and π_1 holds on the intermediate suffixes. The standard temporal operators of LTL can be expressed by $\mathbf{f}$ and $\mathbf{suf}$ (cf. [11]), for instance:

$$\mathbf{n}\,\phi \equiv (\neg \mathit{true})\ \mathbf{suf}\ \phi,\ \phi\,\mathrm{U}\,\psi \equiv \psi \vee (\phi \wedge \phi\ \mathbf{suf}\ \psi),$$

where $\mathbf{n}$ denotes the *next* operator and U denotes the *until* operator of LTL [11]. Intuitively, $\mathbf{n}\,\phi$ holds on a trace when ϕ holds from the second state onward, while $\phi\,\mathrm{U}\,\psi$ holds when ψ eventually holds and ϕ holds on all intermediate steps. Other Boolean connectives, such as $\vee$ and $\rightarrow$, are treated as abbreviations in the standard way. We also use the dual modality $\langle\alpha\rangle\pi$, defined as $\neg[\alpha]\neg\pi$, to state that there exists an execution trace of α satisfying π.

The semantics of PPL is based on *worlds*, which abstract program states. A *trace* tr over a set $\mathcal{S}$ of worlds is a finite non-empty sequence of worlds. Given two traces $tr_1, tr_2 \in \mathcal{S}^*$ with $tr_1 =_{df} s_1 \dots s_n$ and $tr_2 =_{df} t_1 \dots t_m$, their concatenation $tr_1 \cdot tr_2$ is defined as $s_1 \dots s_n t_2 \dots t_m$ when $s_n = t_1$. For a trace tr, let tr_b and tr_e denote its first and last elements. We write $tr_1 \preceq_s tr_2$ when tr_1 is a suffix of tr_2, and $tr_1 \prec_s tr_2$ when it is a proper suffix.

A *Kripke frame* is a pair $(\mathcal{S}, \mathcal{I})$, where $\mathcal{S}$ is a set of worlds and $\mathcal{I}$ interprets each atomic program as a set of length-two traces and each proposition as a set of worlds.

Definition 2.2 (Semantics of PPL). *Given a Kripke frame $(\mathcal{S}, \mathcal{I})$, the semantics of a program and a formula are defined as an extension of function $\mathcal{I}$. It is defined by simultaneous inductions on the syntactic structures of both programs and formulas:*

1. $\mathcal{I}(\phi?) =_{df} \{ss \mid s \in \mathcal{I}(\phi)\}$.
2. $\mathcal{I}(\alpha;\beta) =_{df} \mathcal{I}(\alpha) \cdot \mathcal{I}(\beta)$, *where* $\mathcal{I}(\alpha) \cdot \mathcal{I}(\beta) =_{df} \{tr_1 tr_2 \mid tr_1 \in \mathcal{I}(\alpha), tr_2 \in \mathcal{I}(\beta)\}$.
3. $\mathcal{I}(\alpha \cup \beta) =_{df} \mathcal{I}(\alpha) \cup \mathcal{I}(\beta)$.
4. $\mathcal{I}(\alpha^*) =_{df} \bigcup_{n \geq 0} \mathcal{I}(\alpha^n)$, *where* $\alpha^0 =_{df} true?$, $\alpha^n =_{df} \alpha;\alpha^{n-1}$ $(n \geq 1)$.
5. $\mathcal{I}(true) =_{df} S$.
6. $\mathcal{I}(\neg\phi) =_{df} S \setminus \mathcal{I}(\phi)$.
7. $\mathcal{I}(\phi \wedge \psi) =_{df} \mathcal{I}(\phi) \cap \mathcal{I}(\psi)$.
8. $\mathcal{I}(\mathbf{f}\,\phi) =_{df} \{tr \mid tr_b \in \mathcal{I}(\phi)\}$.
9. $\mathcal{I}(\phi\ \mathbf{suf}\ \psi) =_{df} \{tr \mid \exists tr'.tr' \prec_s tr \wedge tr' \in \mathcal{I}(\psi) \wedge (\forall tr''.tr' \prec_s tr'' \prec_s tr \rightarrow tr' \in \mathcal{I}(\phi))\}$.
10. $\mathcal{I}([\alpha]\phi) =_{df} \{tr \mid for\ all\ tr' \in \mathcal{I}(\alpha), trtr' \in \mathcal{I}(\phi)\}$.

The clauses above follow the intended trace interpretation of regular programs and process formulas. Tests produce stuttering traces when their condition holds; Sequence composes compatible traces; Choice takes union; And iteration takes the union over all finite unfoldings. Boolean connectives are interpreted set-theoretically. The trace operators inspect the beginning and suffixes of a trace, and $[\alpha]\phi$ holds of a trace precisely when appending any trace generated by α yields a trace satisfying ϕ.

Given a Kripke frame $(\mathcal{S}, \mathcal{I})$, a trace $tr \in \mathcal{S}^*$ *satisfies* a formula ϕ, written $tr \models_{(\mathcal{S},\mathcal{I})} \phi$ or simply $tr \models \phi$, if $tr \in \mathcal{I}(\phi)$. State satisfaction is included as the special case of traces of length one.

The proof system of PPL is recalled in Table 1. It consists of the usual propositional dynamic logic (PDL) axioms and rules (cf. [12]), together with the additional rules for process-logic trace operators. For the explanations of these additional rules, one can refer to [11].

The following result is directly from [11].

Proposition 2.1. *The proof system of PPL in Table 1 is sound and complete.*

2.2 First-order Process Logic

First-order process logic (FOPL) enriches PPL by interpreting atomic programs as concrete assignment steps over an arithmetic structure and atomic propositions as first-order predicates over program states. This expressiveness is sufficient to capture a wide range of computational systems: first-order dynamic logic (FODL) underlies verification frameworks for Java [3], hybrid systems [18], and synchronous programs, among others.

When studying FODL, it often considers a first-order structure where first-order quantification is allowed [12]. In this paper, for simplicity of discussion, we consider a simple version of the arithmetic theory over integers, named $\mathfrak{A}$, as the first-order structure for the first-ordered level of process logic. The arithmetic theory over integers is expressive enough to capture most interesting computational structures in computer systems. The result of this paper can also be easily extended to other first-order structures.

No.	Rule or axiom
PDL part	
(1)	$[\alpha](\phi \to \psi) \to ([\alpha]\phi \to [\alpha]\psi)$
(2)	$[\alpha](\phi \wedge \psi) \leftrightarrow [\alpha]\phi \wedge [\alpha]\psi$
(3)	$[\alpha \cup \beta]\phi \leftrightarrow [\alpha]\phi \wedge [\beta]\phi$
(4)	$[\alpha ; \beta]\phi \leftrightarrow [\alpha][\beta]\phi$
(5)	$[\psi?]\phi \leftrightarrow (\psi \to \phi)$
(6)	$[\alpha^*]\phi \leftrightarrow \phi \wedge [\alpha][\alpha^*]\phi$
(7)	$\phi \wedge [\alpha^*](\phi \to [\alpha]\phi) \to [\alpha^*]\phi$
(MP)	$\dfrac{\psi \quad \psi \to \phi}{\phi}\ (MP)$
(Gen)	$\dfrac{\phi}{[\alpha]\phi}\ (Gen)$
PPL-specific part	
(i)	$\mathbf{f}(\phi \vee \psi) \leftrightarrow \mathbf{f}\,\phi \vee \mathbf{f}\,\psi$
(ii)	$\mathbf{f}\,\neg\phi \leftrightarrow \neg\,\mathbf{f}\,\phi$
(iii)	$(\phi\ \mathbf{suf}\ \psi) \vee (\phi\ \mathbf{suf}\ \chi) \leftrightarrow \phi\ \mathbf{suf}\ (\psi \vee \chi)$
(iv)	$(\phi\ \mathbf{suf}\ \psi) \wedge (\chi\ \mathbf{suf}\ \omega) \leftrightarrow (\phi \wedge \chi)\ \mathbf{suf}\ (\psi \wedge \omega) \vee (\phi \wedge \chi)\ \mathbf{suf}\ (\phi \wedge \omega \wedge \phi\ \mathbf{suf}\ \psi) \vee (\phi \wedge \chi)\ \mathbf{suf}\ (\psi \wedge \chi \wedge \chi\ \mathbf{suf}\ \omega)$
(v)	$\neg(\phi\ \mathbf{suf}\ \psi) \leftrightarrow \neg(\mathit{true}\ \mathbf{suf}\ \psi) \vee (\neg\psi)\ \mathbf{suf}\ (\neg\phi \wedge \neg\psi)$
(vi)	$\phi\ \mathbf{suf}\ \psi \leftrightarrow \mathbf{n}\,\psi \vee \mathbf{n}(\phi \wedge (\phi\ \mathbf{suf}\ \psi))$
(vii)	$\phi\ \mathbf{suf}\ (\phi \wedge (\phi\ \mathbf{suf}\ \psi)) \leftrightarrow \mathbf{n}(\phi \wedge (\phi\ \mathbf{suf}\ \psi))$
(viii)	$\phi\ \mathbf{suf}\ (\phi \wedge (\phi\ \mathbf{suf}\ \psi)) \leftrightarrow \phi\ \mathbf{suf}\ (\phi \wedge \mathbf{n}\,\psi)$
(ix)	$\mathbf{f}\,\mathbf{n}\ \mathit{true} \leftrightarrow \mathit{false}$
(x)	$\neg\phi \wedge \mathbf{f}\,\phi \to \mathbf{n}\ \mathit{true}$
(xi)	$p \leftrightarrow \mathbf{f}\,p$, where p is an atomic proposition
(xii)	$\mathbf{f}\,\phi \wedge \langle\alpha\rangle\psi \leftrightarrow \langle\alpha\rangle(\mathbf{f}\,\phi \wedge \psi)$
(xiii)	$\mathbf{n}\langle\alpha\rangle\phi \leftrightarrow \mathbf{n}\ \mathit{true} \wedge \langle\alpha\rangle\,\bar{\mathbf{n}}\,\phi$
(xiv)	$\langle\alpha\rangle\mathit{true} \to \mathbf{fin}$
(xv)	$((\mathbf{n}\,\phi \to \phi)\ \mathbf{suf}\ \phi) \to \mathbf{n}\,\phi$

Table 1. Axioms and rules of propositional process logic.

In the following of this paper, let *Var* be the set of variables over the integer set $\mathbb{Z}$. We use letters u, v to range over the variables of *Var*. An *arithmetic expression* (simply "expression") e is defined recursively as follows:

$$e =_{df} u \mid n \mid e \bowtie e,$$

where $u \in Var$, $n \in \mathbb{Z}$ is a constant, $\bowtie \in \{+, -, *, \div\}$ represents the usual operators in $\mathfrak{A}$. In $\mathfrak{A}$, a world is a mapping from *Var* to $\mathbb{Z}$. Given a world $s : Var \to \mathbb{Z}$, $s[u \mapsto i]$ returns a mapping that maps u to value $i \in \mathbb{Z}$ and maps any other variable v to the value $s(v)$. Given an expression e, the *evaluation* of e by s, denoted by $s(e)$, is defined in the usual way: $s(n) =_{df} n$; $s(e_1 \bowtie e_2) =_{df} s(e_1) \bowtie s(e_2)$ for any e_1, e_2.

A formula in first-order process logic (FOPL) is defined just as in Definition 2.1, except that an atomic program is an assignment $u := e$, where $u \in Var$ is a variable and e is an expression; An atomic formula p is a predicate of the form $e_1 \triangle e_2$, where $\triangle \in \{<, \leq, >, \geq, =\}$ are the usual binary integer relations; Given an FOPL formula ϕ, $\forall u.\phi(u)$ is also a formula quantified by operator $\forall$ on variable $u \in Var$. The evaluation of p by s is defined in the usual way: $s(e_1 \triangle e_2) =_{df} s(e_1) \triangle s(e_2)$.

The *substitution* in FOPL is defined in a usual way. Given a formula ϕ, $\phi[e/u]$ denotes the result obtained by replacing each free occurrence (not bound by any quantifier) of variable u with term e in ϕ. We always assume that a substitution is *admissible*, in the sense that the substitution does not affect the freeness of the other existing variables in ϕ except u.

The special theory $\mathfrak{A}$ fixes the Kripke frame in FOPL, namely $(\mathcal{S}_{fo}, \mathcal{I}_{fo})$, where $\mathcal{S}_{fo}$ is the set of all worlds in $\mathfrak{A}$, and $\mathcal{I}_{fo}$ interprets an assignment and a proposition in $\mathfrak{A}$ as follows:

1. $\mathcal{I}_{fo}(u := e) = \{ss' \mid s' = s[u \mapsto s(e)]\}$.
2. $\mathcal{I}_{fo}(p) = \{s \mid s(p) \text{ is true in theory } \mathfrak{A}\}$.

The semantics of FOPL is defined by extending the definitions of $(\mathcal{S}_{fo}, \mathcal{I}_{fo})$ to other programs and formulas, which is defined just as in Definition 2.2, except that for a quantified formula of the form $\forall u.\phi$, we have

$$\mathcal{I}_{fo}(\forall u.\phi) = \{s \mid \forall i \in \mathbb{Z}.s[u \mapsto i](\phi)\}.$$

3 Overview

Having recalled the basic definitions of process logic, we now step back and ask: *why do we need a new proof framework, and how does our approach work?* This section gives an informal tour of the main ideas before the technical development begins.

Section 3.1 motivates our work by showing that classical and trace-based program logics can be embedded into process logic, making a complete proof system for it broadly useful. Section 3.2 introduces the core idea of the cyclic labelled approach and outlines the soundness and completeness arguments for the proposed labelled systems.

3.1 Motivations of Process Logic

Process logic is highly expressive: its trace-based formulas subsume classical Hoare triples, dynamic-logic modalities, and the richer LTL-flavoured properties studied in recent traced program logics. Providing a *complete* proof system for it is therefore both theoretically significant and broadly applicable, yet non-trivial — especially at the first-order level. In the subsections below we make this motivation concrete by showing how the key logics from the literature each embed into process logic.

3.1.1 From State Properties to Trace Properties: Hoare Logic and Dynamic Logic. Because process logic evaluates formulas over execution *traces*, it is strictly more expressive than purely state-based logics. In particular, it subsumes the classical Hoare-logic specification $\{\phi\}\alpha\{\psi\}$, which asserts that if a program state satisfies the precondition ϕ then, after executing program α, the resulting state satisfies the postcondition ψ.

In process logic, this correctness condition is captured by the formula

$$\phi \to [\alpha]\Box(Ł_0 \to \psi),$$

where the auxiliary abbreviations are defined as follows [11]:

$$\Box\varphi =_{df} \neg(\Diamond\neg\varphi),$$
$$\Diamond\varphi =_{df} \mathit{true}\,\mathrm{U}\,\varphi,$$
$$Ł_0 =_{df} \neg\,\mathbf{n}\,\mathit{true}.$$

Here, $\Box\varphi$ ("always φ") means that φ holds at *every* state along the current trace; $\Diamond\varphi$ ("eventually φ") means that φ holds at *some* state along the trace; and $Ł_0$ ("last state") is a formula that holds precisely at the *final* state of a trace, since $\neg\,\mathbf{n}\,\mathit{true}$ is satisfied exactly when there is no next step. Thus $\Box(Ł_0 \to \psi)$ asserts that at the last state of every execution trace of α the formula ψ holds — which is exactly the Hoare postcondition.

Process logic also subsumes classical dynamic logic. In dynamic logic, every formula is a *state formula* — it is satisfied by a single world, not by a trace. The box modality $[\alpha]\phi$ therefore asserts that the world obtained after executing α (in any possible way) satisfies ϕ. Because process logic evaluates formulas over traces rather than over single states, the equivalent process-logic statement must single out the *last* state of the execution trace. This is achieved by the formula

$$[\alpha]\Box(Ł_0 \to \phi),$$

which, by the same reasoning as above, says that at the final state of every execution trace of α the formula ϕ holds.

3.1.2 Existing Trace-Based Dynamic Logics and Their Limitations. Process logic also subsumes the various trace-semantic-based dynamic logics discussed in Section 1. For each, we briefly explain its core idea and show how its key formulas and rules can be expressed in process logic. We also highlight the limitations of each approach, which motivates our own labelled framework introduced in Section 3.2.

Dynamic Logic with Trace Modalities (DLT). DLT [4] extends FODL with a trace modality $[|\cdot|]$ that captures the intermediate worlds visited during program execution. In DLT, the underlying program model consists of deterministic while programs, and the proof system is proved to be complete for this class [4]. The formula $[|\alpha|]\phi$ means that ϕ (a state formula) holds at *every* intermediate world along every execution trace of α. In process logic, this is captured precisely by

$$[\alpha]\Box\phi,$$

using the "always" operator $\Box$ introduced earlier.

For sequential programs, [4] proposes the following compositional rule, which we call $([;]-\Box)$:

$$\frac{[|\alpha|]\phi \wedge [|\alpha|][|\beta|]\phi}{[|\alpha\,;\beta|]\phi}$$

Expressed as a process-logic axiom, this rule becomes:

$$[\alpha\,;\beta]\Box\phi \leftrightarrow [\alpha]\Box\phi \wedge [\alpha](\mathbf{last}[\beta]\Box\phi),$$

where $\mathbf{last}\,\varphi =_{df} \Diamond(\varphi \wedge Ł_0)$ for any formula φ. Intuitively, $\mathbf{last}\,\varphi$ means that φ holds *at the last state of the current trace*: $\Diamond(\varphi \wedge Ł_0)$ requires that there is some state on the trace that is simultaneously the final state ($Ł_0$) and satisfies φ. The axiom then says: every trace of $\alpha\,;\beta$ satisfies $\Box\phi$ if and only if every trace of α already satisfies $\Box\phi$, *and* at the end of every trace of α, every continuation trace of β also satisfies $\Box\phi$. In other words, the axiom splits the sequential execution into two parts and ensures that ϕ is maintained throughout both segments. In process logic, this rule is more complex than the simpler rule (4) of Table 1 for handling sequential programs.

The subsequent work [17] proposes the logic DTL_d, which adopts the trace modalities of DLT as an extension to differential dynamic logic [16], applying them to the more general regular programs. However, it remains open whether the system fragment of DLT in DTL_d is complete for this broader program class. Moreover, [22] shows that for regular programs the direction $[\alpha\,;\beta]\Box\phi \to [\alpha]\Box\phi \wedge [\alpha](\mathbf{last}[\beta]\Box\phi)$ does not hold: a test in $\alpha\,;\beta$ may block a trace of α from continuing as a trace of β, thereby invalidating the compositional split. Consequently, rule $([;]-\Box)$ is too strong and is thus insufficient to derive all valid trace properties for arbitrary regular programs.

Differential Temporal Dynamic Logic DTL_d^2. It was later observed in [17] that rule $([;]-\Box)$ cannot be directly applied to reasoning about $[\alpha]\Diamond\phi$ for the "eventually" operator $\Diamond$: the rule breaks down because distributing the $\Diamond$ modality over a sequential composition does not follow the same pattern as $\Box$. To handle "eventually" formulas, [13] introduces the logic DTL_d^2, which enriches formula $[\alpha]\Diamond\phi$ to the richer form $[\alpha](\psi \sqcup \Diamond\phi)$ and proposes a compositional

rule $([;]-\Diamond)$:

$$\frac{[\alpha]([\beta](\psi \sqcup \Diamond\phi) \sqcup \Diamond\phi)}{[\alpha;\beta](\psi \sqcup \Diamond\phi)}.$$

In process logic, the formula $[\alpha](\psi \sqcup \Diamond\phi)$ is captured as

$$[\alpha](\psi \vee \Diamond\phi),$$

which means that each trace of α satisfies either ψ or $\Diamond\phi$. And the rule $([;]-\Diamond)$ becomes the axiom:

$$[\alpha;\beta](\psi \vee \Diamond\phi) \leftrightarrow [\alpha](\mathbf{last}([\beta](\psi \vee \Diamond\phi)) \vee \Diamond\phi).$$

Informally, the axiom decomposes the "eventually" property of the sequential program $\alpha;\beta$: the combined execution satisfies $\psi \vee \Diamond\phi$ if and only if, at the end of every α-trace, either (i) $\Diamond\phi$ has already been witnessed within that α-trace, or (ii) the subsequent β-execution will witness $\psi \vee \Diamond\phi$ from that terminal state.

A later work [1] proves that DTL^2_d is complete for deterministic program behaviours, i.e., when the choice operator $\cup$ in a regular program does not yield non-deterministic behaviours, e.g., $\phi?;a \cup \neg\phi?;b$ for any ϕ, a, b. However, the completeness for general regular programs with genuine nondeterminism remains open; moreover, it is not clear whether any fixed set of compositional rules suffices to handle all LTL operators uniformly.

Dynamic Trace Logic (DTL). An earlier work [2] proposes another variant of trace-based dynamic logic called dynamic trace logic (DTL). The program models of DTL are restricted to while programs, which are deterministic: given a fixed starting world, there is only one possible execution trace. Under this restriction, the proof system of DTL can derive every valid formula $[\alpha]\phi$ for an *arbitrary* LTL formula ϕ. The key insight is that compositional rules for a single deterministic trace work uniformly for all types of temporal properties. However, this argument breaks down for regular programs in general, where nondeterminism allows multiple possible execution traces and the compositional rules must handle branching.

3.2 Labelled System & Our Approach

3.2.1 An Labelled Approach for Process Logic. We adopt an alternative approach to provide a complete logical framework for process logic at both the propositional and first-order levels. Our approach draws on the rich tradition of *labelled proof systems* for modal and dynamic logics [9, 15], which augment formulas with explicit structural information — called *labels* — that records the relational or semantic context in which a formula is evaluated. By making this context part of the formula itself, labelled systems often admit cleaner, more modular inference rules than their unlabelled counterparts.

Following the main idea from prior work on labelled dynamic-logic systems [9, 21, 23], we introduce a label σ to capture the current execution-trace information accumulated during a derivation. A labelled process-logic formula takes the form $\sigma : [\alpha]\phi$, meaning that under the trace information recorded in σ, every execution trace of α satisfies formula ϕ. This labelled form is strictly more general than the plain formula $[\alpha]\phi$: when σ is a free trace variable X with no additional constraints, $\sigma : [\alpha]\phi$ and $[\alpha]\phi$ share the same validity.

To see why labels help, consider the formula

$$\{x := 1\} : [x := 2; x := 3]\Box(x \geq 0),$$

which means that given the execution trace information $\{x := 1\}$ (recording that x was set to 1 at the first step), every continuation via program $x := 2; x := 3$ produces a whole trace satisfying $\Box(x \geq 0)$. By applying a labelled version of

the sequential-composition rule, we can unfold this derivation step by step, accumulating execution information in the label:

$$\dfrac{\dfrac{\dfrac{\dfrac{\{x := 1\} \diamond \{x := 2\} \diamond \{x := 3\} : \Box(x \geq 0)}{\{x := 1\} \diamond \{x := 2\} : [x := 3]\Box(x \geq 0)}}{\{x := 1\} : [x := 2][x := 3]\Box(x \geq 0)}}{\{x := 1\} : [x := 2 ; x := 3]\Box(x \geq 0)} \quad ,$$

where the label $\{x := 1\} \diamond \{x := 2\} \diamond \{x := 3\}$ records the full execution history. Once all programs have been unfolded, the verification goal reduces to checking whether this concrete label satisfies the temporal formula $\Box(x \geq 0)$ — a much simpler task. This contrasts with the traditional approach using rule $([;]-\Box)$, which would split $[x := 2 ; x := 3]\Box(x \geq 0)$ into separate sub-goals $[x := 2]\Box(x \geq 0)$ and $[x := 3](\textbf{last}([x := 3]\Box(x \geq 0)))$ that must be handled independently.

The formal construction of the labelled systems for PPL and FOPL — namely G3PPL and G3FOPL — is carried out in Sections 4.1, 4.2, 7.1, and 7.2, respectively.

Similar to [9, 21], our labelled system for process logic does not directly support reasoning about iterative regular programs: when unwrapping the iterative structure, the same program form repeats during a derivation, producing an infinite derivation. The following derivation illustrates this:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dots}{\{x := n\} \diamond \dots \diamond \{x := 1\} \diamond X : [(x := 1)^*]\mathit{true}}}{\vdots}}{\{x := 1\} \diamond X : [(x := 1)^*]\mathit{true}}}{X : [(x := 1)^*]\mathit{true}} \quad ,$$

where at each derivation step, $x := 1$ is symbolically executed and $(x := 1)^*$ is transformed into itself, starting a new loop.

To solve this problem, we adapt the cyclic proof approach [6, 7, 20] to our labelled system. A cyclic proof (Definition 4.7) admits a certain type of derivation branches whose leaf nodes either terminate (closed by an axiom) or can "backlink" to some identified ancestor along the same branch. For instance, formula $X : [(x := 1)^*]\mathit{true}$ can be closed by a suitable variable substitution to form a leaf node $Y : [(x := 1)^*]\mathit{true}$ that backlinks to the root $X : [(x := 1)^*]\mathit{true}$ (modulo free-variable renaming):

$$\dfrac{\dfrac{Y : [(x := 1)^*]\mathit{true}}{\{x := 1\} \diamond X : [(x := 1)^*]\mathit{true}}\ (\mathit{sub})}{X : [(x := 1)^*]\mathit{true}} \quad .$$

We discuss the condition — called "cyclic condition" (Definition 4.6) — under which such a cyclic derivation constitutes a legal cyclic proof that leads to a valid conclusion.

Sections 4.3 and 7.2 present the details of the cyclic theories for G3PPL and G3FOPL, respectively.

3.2.2 Analysis of Soundness and Completeness. As the major theoretical results, we fully prove the soundness and completeness of the proposed labelled systems for process logic.

Soundness. We establish soundness at two levels. First, each individual rule of the labelled system is locally sound (Theorem 4.1). Local soundness alone does not suffice, however, because a cyclic derivation may contain infinite derivation branches, and local soundness does not directly guarantee that such a branch leads to a valid conclusion. We therefore establish a second, global soundness result: every cyclic derivation satisfying the cyclic condition yields a

valid conclusion. This is discussed in detail in Section 5. For both G3PPL and G3FOPL, the soundness arguments follow the main strategy from [21], adapted with non-trivial technical details specific to the trace semantics of process logic.

Completeness. We establish completeness using different proof methods for G3PPL and G3FOPL, discussed in Sections 6 and 7.3 respectively. For G3PPL, the key idea is to show that the labelled system can simulate every axiom and rule of the established Hilbert-style proof system for PPL displayed in Table 1. We present the technical details for the critical "Necessitation" rule and the PDL axioms (rules (1)–(7), (MP), (Gen)), leaving most of the proofs for the process-logic-specific axioms (rules (i)–(xv)) to the appendix. For G3FOPL, we adapt the classical completeness argument for FODL [10, 12] to the labelled setting with trace semantics, providing the necessary technical modifications.

4 Labelled Propositional Process Logic

This section develops the labelled propositional process logic and its proof system. We first define *labels* and *labelled formulas*, then present the labelled sequent rules and the cyclic proof condition required for regular-program iteration.

4.1 Labelled PPL Formulas

Definition 4.1 (Labels of PPL). *A label σ is a trace defined as follows:*

$$\sigma =_{df} \varepsilon \mid x \mid X \mid \sigma \diamond \sigma,$$

where x is a label variable for worlds; X is a label variable for traces of worlds; $\diamond$ is a concatenation operator between label variables; ε is called the "empty label", which satisfies that $\sigma \diamond \varepsilon = \varepsilon \diamond \sigma = \sigma$ for any label σ.

Use $\mathbf{L}$ to denote the set of all labels. Use $WVar$ to express the set of all label variables for worlds; use $TVar$ to express the set of all label variables for traces. Define $LVar =_{df} WVar \cup TVar$ be the set of all label variables.

The *substitution* on labels is defined in a natural way: for a label σ, $\sigma[\sigma'/U]$ returns the label by replacing each free occurrence of variable $U \in LVar$ with label σ'. Since no quantifiers for label variables exist, all occurrences of label variables are free.

Definition 4.2 (Label Mappings of PPL). *Given a Kripke frame $(\mathcal{S}, \mathcal{I})$, a world mapping $\mathfrak{w}$ in PPL is a function from $WVar$ to $\mathcal{S}$.*

A label mapping $\mathfrak{m}$ is a function from $\mathbf{L}$ to the set $\mathcal{S}^*$ of world traces, defined as follows:

i $\mathfrak{m}$ *maps a variable $X \in TVar$ to a trace of worlds in $\mathcal{S}^*$.*
ii $\mathfrak{m}(\varepsilon) =_{df} \epsilon$.
iii $\mathfrak{m}(x) =_{df} \mathfrak{w}(x)$.
iv $\mathfrak{m}(\sigma_1 \diamond \sigma_2) =_{df} \mathfrak{m}(\sigma_1) \cdot \mathfrak{m}(\sigma_2)$.

Denote the set of all label mappings (w.r.t. $(\mathcal{S}, \mathcal{I})$) as $\mathbf{M}(\mathcal{S}, \mathcal{I})$ (simply $\mathbf{M}$).

Definition 4.3 (Labelled PPL Formulas). *A "labelled PPL formula" is of the form:*

$$\sigma : \phi,$$

where $\sigma \in \mathbf{L}$, ϕ is a PPL formula.

Definition 4.4 (Semantics of Labelled PPL Formulas). *Given a Kripke frame $(\mathcal{S}, \mathcal{I})$ and a set $\mathbf{M}$ of label mappings w.r.t. $(\mathcal{S}, \mathcal{I})$, the satisfaction relation $\mathfrak{m} \models_{\mathbf{M}} \sigma : \phi$ of a labeled formula $\sigma : \phi$ by a label mapping $\mathfrak{m}$ (w.r.t. $\mathbf{M}$) is defined*

such that

$$\mathfrak{m} \models_{\mathbf{M}} \sigma : \phi, \ \textit{if}\ \mathfrak{m}(\sigma) \models_{(\mathcal{S},\mathcal{I})} \phi.$$

Simply write $\models_{\mathbf{M}}$ as $\models$ if from the context $\mathbf{M}$ and $(\mathcal{S}, \mathcal{I})$ are clear.

4.2 A Proof System for Labelled PPL

A *sequent* is a logical argumentation of the form:

$$\Gamma \Rightarrow \Delta,$$

where Γ and Δ are finite multi-sets of formulas, called the *left side* and the *right side* of the sequent respectively. We use dot $\cdot$ to express Γ or Δ when they are empty sets. Intuitively, a sequent $\Gamma \Rightarrow \Delta$ means that if all formulas in Γ hold, then one of formulas in Δ holds. We use ν to represent a sequent. An *inference rule* is of the form

$$\frac{\nu_1 \quad ... \quad \nu_n}{\nu},$$

with the premises $\nu_1, ..., \nu_n$ and the conclusion ν. The semantics of the rule is that the validity of sequents $\nu_1, ..., \nu_n$ implies the validity of sequent ν. We call a formula pair (τ_1, τ_2) a *conclusion-premise* (CP) pair if τ_1 and τ_2 belong to nodes ν, ν_i for some $1 \leq i \leq n$ respectively. In this paper, we use a double-lined inference form:

$$\frac{\phi_1 \quad ... \quad \phi_n}{\phi}$$

as a shorthand to represent both rules

$$\frac{\Gamma \Rightarrow \phi_1, \Delta \quad ... \quad \Gamma \Rightarrow \phi_n, \Delta}{\Gamma \Rightarrow \phi, \Delta} \quad \text{and} \quad \frac{\Gamma, \phi_1 \Rightarrow \Delta \quad ... \quad \Gamma, \phi_n \Rightarrow \Delta}{\Gamma, \phi \Rightarrow \Delta},$$

provided any context Γ and Δ.

Given a proof system, i.e., a set of inference rules, a *proof* of a sequent is a finite *proof tree* formed by deriving the sequent by applying these rules backwardly (i.e., from the conclusions to the premises). In this tree, each node is a sequent; the root node is the sequent itself (the conclusion); each leaf node is *terminal* in the sense that it is the conclusion of an axiom. We say that a sequent can be proved if there is a proof for it.

The proof system of labelled PPL is displayed in Table 2, 3 and 4, which consists of 3 parts: the rules for regular programs (Table 2), which are extended from the G3 sequent calculus for modal logics [15], the rules for trace formulas (Table 3), which are special in process logic, and the rules for labelled version of proposition logical formulas (Table 4), which are augmented based on the classical sequent calculus for propositional logic. We call the resulting proof system G3PPL.

In the displayed rules of G3PPL, a *target pair* is the CP pair explicitly transformed in the rule, apart from the unchanged contexts Γ and Δ. E.g., $(\sigma \diamond x : [a]\phi, \sigma \diamond x \diamond y : \phi)$ is a target pair of an instance of rule $([a]R)$, where $\sigma \diamond x \diamond y : \phi$ is transformed from $\sigma \diamond x : [a]\phi$. While $(\sigma \diamond x : [a]\phi, x\, \mathrm{R}_a\, y)$ is not a target pair since $x\, \mathrm{R}_a\, y$ is new generated, not transformed from $\sigma \diamond x : [a]\phi$. The structural rules (sub), (ax), and (cut) do not have target pairs.

The rules for regular programs in Table 2 follow the standard labelled-sequent treatment of modal operators [15], adapted to the trace semantics of PPL. Rules $([a]R)$ and $([a]L)$ handle the box modality over atomic programs: to prove $\sigma \diamond x : [a]\phi$, rule $([a]R)$ introduces a fresh world variable y representing the successor of x under action a, reducing the goal to $\sigma \diamond x \diamond y : \phi$ with the relational atom xR_ay in the antecedent; rule $([a]L)$ uses an existing relational atom to consume the modality from the left. Rules $([\phi?]R)$ and $([\phi?]L)$ handle tests: a test $\psi?$ produces a stuttering trace, so the rule appends x to itself and requires the test condition to be satisfied. Rules $([;])$, $([\cup])$, and $([*])$ unfold sequential

Rule	Schematic form
$([a]R)$	$\dfrac{\Gamma, x\,\mathrm{R}_a\,y \Rightarrow \sigma \diamond x \diamond y : \phi, \Delta}{\Gamma \Rightarrow \sigma \diamond x : [a]\phi, \Delta}$, where $y \in \mathit{WVar}$ is fresh
$([a]L)$	$\dfrac{\Gamma, \sigma \diamond x \diamond y : \phi \Rightarrow \Delta}{\Gamma, \sigma \diamond x : [a]\phi, x\,\mathrm{R}_a\,y \Rightarrow \Delta}$, for some $y \in \mathit{WVar}$
$([\phi?]R)$	$\dfrac{\Gamma, x \diamond x : \psi \Rightarrow \sigma \diamond x \diamond x : \phi, \Delta}{\Gamma \Rightarrow \sigma \diamond x : [\psi?]\phi, \Delta}$
$([\phi?]L)$	$\dfrac{\Gamma, \sigma \diamond x \diamond x : \phi \Rightarrow \Delta}{\Gamma, \sigma \diamond x : [\psi?]\phi, x \diamond x : \psi \Rightarrow \Delta}$
$([;])$	$\dfrac{\sigma : [\alpha][\beta]\phi}{\sigma : [\alpha\,;\beta]\phi}$
$([\cup])$	$\dfrac{\sigma : [\alpha]\phi \wedge [\beta]\phi}{\sigma : [\alpha \cup \beta]\phi}$
$([*])$	$\dfrac{\sigma : [\mathit{true}? \cup \alpha\,;\alpha^*]\phi}{\sigma : [\alpha^*]\phi}$
(sub)	$\dfrac{\Gamma \Rightarrow \Delta}{\Gamma[\sigma/U] \Rightarrow \Delta[\sigma/U]}$, where $U \in \mathit{LVar}$ is fresh w.r.t. Γ and Δ

Table 2. Rules of G3PPL for regular programs and label substitution.

Rule	Schematic form
(p)	$\dfrac{x : p}{x \diamond \sigma : p}$
$(\mathbf{f})$	$\dfrac{x : \phi}{x \diamond \sigma : \mathbf{f}\,\phi}$
$(\mathbf{suf})$	$\dfrac{\sigma : \psi \vee (\phi \wedge \phi\ \mathbf{suf}\ \psi)}{x \diamond \sigma : \phi\ \mathbf{suf}\ \psi}$
$(\mathbf{suf}\ \ x)$	$\dfrac{}{\Gamma, x : \phi\ \mathbf{suf}\ \psi \Rightarrow \Delta}$

Table 3. Rules of G3PPL for trace formulas.

composition, choice, and iteration, respectively, matching the standard PDL axioms (Table 1). Rule (sub) substitutes a fresh label variable U with a concrete label σ throughout a sequent; it is the key mechanism for constructing back-links in cyclic proofs.

The rules for trace formulas in Table 3 capture the semantics of the temporal operators in process logic. Rule (p) states that an atomic proposition p evaluated on a trace $x \diamond \sigma$ depends only on the first world x: if $x : p$ holds, then so does $x \diamond \sigma : p$ for any σ. Rule $(\mathbf{f})$ similarly reduces the evaluation of $\mathbf{f}\,\phi$ on $x \diamond \sigma$ to the evaluation of ϕ at x alone. Rule $(\mathbf{suf})$ is the key rule for the suffix connective: on a trace $x \diamond \sigma$, the formula $\phi\ \mathbf{suf}\ \psi$ reduces to the disjunction $\psi \vee (\phi \wedge \phi\ \mathbf{suf}\ \psi)$ on σ, reflecting that either the suffix σ already satisfies ψ, or it satisfies ϕ and the same condition propagates along σ. Finally, rule $(\mathbf{suf}\ \ x)$ is an axiom that closes a branch when the trace reduces to a single world variable x: since x has no proper suffix, $x : \phi\ \mathbf{suf}\ \psi$ is vacuously false and the branch terminates.

Rule	Schematic form
(ax)	$\dfrac{}{\Gamma, \sigma : \phi \Rightarrow \sigma : \phi, \Delta}\ (ax)$
(cut)	$\dfrac{\Gamma \Rightarrow \sigma : \phi, \Delta \quad \Gamma, \sigma : \phi \Rightarrow \Delta}{\Gamma \Rightarrow \Delta}\ (cut)$
(wk)	$\dfrac{\Gamma \Rightarrow \Delta}{\sigma : \phi}\ (wk)$
(ctr)	$\dfrac{\sigma : \phi, \sigma : \phi}{\sigma : \phi}\ (ctr)$
$(\neg R)$	$\dfrac{\Gamma, \sigma : \phi \Rightarrow \Delta}{\Gamma \Rightarrow \sigma : \neg\phi, \Delta}\ (\neg R)$
$(\neg L)$	$\dfrac{\Gamma \Rightarrow \sigma : \phi, \Delta}{\Gamma, \sigma : \neg\phi \Rightarrow \Delta}\ (\neg L)$
$(\wedge R)$	$\dfrac{\Gamma \Rightarrow \sigma : \phi, \Delta \quad \Gamma \Rightarrow \sigma : \psi, \Delta}{\Gamma \Rightarrow \sigma : \phi \wedge \psi, \Delta}\ (\wedge R)$
$(\wedge L)$	$\dfrac{\Gamma, \sigma : \phi, \sigma : \psi \Rightarrow \Delta}{\Gamma, \sigma : \phi \wedge \psi \Rightarrow \Delta}\ (\wedge L)$

Table 4. Primitive propositional and structural rules of G3PPL.

Rule	Schematic form
$(\vee R)$	$\dfrac{\Gamma \Rightarrow \sigma : \phi, \sigma : \psi, \Delta}{\Gamma \Rightarrow \sigma : \phi \vee \psi, \Delta}\ (\vee R)$
$(\vee L)$	$\dfrac{\Gamma, \sigma : \phi \Rightarrow \Delta \quad \Gamma, \sigma : \psi \Rightarrow \Delta}{\Gamma, \sigma : \phi \vee \psi \Rightarrow \Delta}\ (\vee L)$
$(\rightarrow R)$	$\dfrac{\Gamma, \sigma : \phi \Rightarrow \sigma : \psi, \Delta}{\Gamma \Rightarrow \sigma : \phi \rightarrow \psi, \Delta}\ (\rightarrow R)$
$(\rightarrow L)$	$\dfrac{\Gamma \Rightarrow \sigma : \phi, \Delta \quad \Gamma, \sigma : \psi \Rightarrow \Delta}{\Gamma, \sigma : \phi \rightarrow \psi \Rightarrow \Delta}\ (\rightarrow L)$

Table 5. Derived propositional rules used in G3PPL.

The rules for labelled propositional logical formulas (Table 4) have the same meaning as their unlabelled counterparts for the classical propositional logic.

Soundness of Rules in G3PPL is stated as the following theorem.

Theorem 4.1. *Each rule in Tables 2, 3, and 4 for G3PPL is sound.*

The proof directly follows from the semantics of labelled PPL formulas. We omit it in this paper.

4.3 Cyclic Proof Structure of G3PPL

Due to the iterative programs in regular programs, the derivation of a formula in G3PPL does not always terminate, as the same formula form may appear repeatedly in a derivation. Here is an example:

$$\dfrac{\dfrac{\dfrac{\cdot \Rightarrow x : [a^*]p}{\cdot \Rightarrow x : [a^*]p, x : [true?]p}\ (wk) \quad \dfrac{\cdot \Rightarrow x : [a^*]p}{\cdot \Rightarrow x : [a^*]p, x : [a\,;a^*]p}\ (wk)}{\cdot \Rightarrow x : [a^*]p, x : [a^*]p}\ ([*],[\cup],\wedge R)}{\cdot \Rightarrow x : [a^*]p}\ (ctr)$$

In this example, using the rules in G3PPL yields the repeated root node in two branches. Continuing using the same rules lead to infinite derivation paths.

In cyclic proof approach, a *preproof* is a finite proof tree in which there exist a special type of non-terminal leaf nodes, called *buds* (cf. [7]). A bud is identical to one of its ancestors in its syntactic form in the tree. A bud and its one of its identical ancestors together is called a *back-link*. A *derivation path* in a proof tree is a sequence of nodes $v_1v_2...v_m...$ $(m \geq 1)$ starting from the root node v_1, where each node pair (v_i, v_{i+1}) $(i \geq 1)$ is an instance of a rule. In a preproof, there always exists an infinite derivation path over one or more derivation branches which contain back-links.

A preproof may not be a valid proof. As in the example shown above, formula $x : [a^*]p$ is concluded in the preproof, but it may not be true under certain Kripke frames (e.g. $x : [a^*]\mathit{false}$). This induces that we need to specify a particular *cyclic condition* for G3PPL, to ensure that a preproof must conclude a sound conclusion.

Below we define the notion of derivation traces in G3PPL, then introduce the progressive derivation traces for G3PPL. Based on these concepts, we finally give the cyclic condition for G3PPL in Definition 4.7.

Definition 4.5 (Derivation Traces). *A "derivation trace" over a derivation path $\mu_1\mu_2...\mu_k v_1v_2...v_m...$ $(k \geq 0, m \geq 1)$ is a sequence $\tau_1\tau_2...\tau_m...$ of formulas, where each τ_i $(1 \leq i \leq m)$ is a formula in node v_i. Each CP pair (τ_i, τ_{i+1}) $(i \geq 1)$ of derivation (v_i, v_{i+1}) satisfies special conditions as follows according to (v_i, v_{i+1}) being the different instances of rules from G3PPL:*

1. *If (v_i, v_{i+1}) is an instance of rule $([a]R)$, $([a]L)$, $([\phi?R])$, $([\phi?L])$, $(\neg R)$, $([;])$, $[\cup]$, $[*]$, (p), $(\mathbf{f})$, $(\mathbf{suf})$, $(\mathbf{suf}\ x)$, $(\neg L)$, $(\wedge R)$ or $(\wedge L)$, then either (τ_i, τ_{i+1}) is a target pair of the rule, or $\tau_i = \tau_{i+1}$;*
2. *If (v_i, v_{i+1}) is an instance of rule (sub), then $\tau_i = sub(\sigma) : \phi$ and $\tau_{i+1} = \sigma : \phi$ for some $\sigma \in \mathbf{L}$ and formula ϕ;*
3. *If (v_i, v_{i+1}) is an instance of the other rules (that do not have target pairs), then $\tau_i = \tau_{i+1}$.*

Definition 4.6 (Progressive Derivation Traces of G3PPL). *In a preproof of system G3PPL, given a derivation trace $\tau_1\tau_2...\tau_m...$ over a derivation path $...v_1v_2...v_m...$ $(m \geq 1)$ starting from τ_1 in node v_1, a CP pair (τ_i, τ_{i+1}) $(1 \leq i \leq m)$ of derivation (v_i, v_{i+1}) is called a "progressive step", if (τ_i, τ_{i+1}) is one of the following CP pairs for different situations of (v_i, v_{i+1}):*

1. *when (v_i, v_{i+1}) is an instance of rule $([a]R)$:*

$$\dfrac{\Gamma, xR_ay \Rightarrow \sigma \diamond x \diamond y : \phi, \Delta}{\Gamma \Rightarrow \sigma \diamond x : [a]\phi, \Delta,}\ ,$$

   then (τ_i, τ_{i+1}) is the pair $(\sigma \diamond x : [a]\phi, \sigma \diamond x \diamond y : \phi)$.
2. *when (v_i, v_{i+1}) is an instance of rule $([\phi?]R)$:*

$$\dfrac{\Gamma, x \diamond x : \psi \Rightarrow \sigma \diamond x \diamond x : \phi, \Delta}{\Gamma \Rightarrow \sigma \diamond x : [\psi?]\phi, \Delta}\ ([\phi?R]),$$

then (τ_i, τ_{i+1}) *is the pair* $(\sigma \diamond x : [\psi?]\phi, \sigma \diamond x \diamond x : \phi)$.

3. *when* (v_i, v_{i+1}) *is an instance of rule* (**suf**)*:*

$$\frac{\sigma : \psi \vee (\phi \wedge \phi \,\mathbf{suf}\, \psi)}{x \diamond \sigma : \phi \,\mathbf{suf}\, \psi}\ (\mathbf{suf}),$$

then (τ_i, τ_{i+1}) *is the pair* $(x \diamond \sigma : \phi \,\mathbf{suf}\, \psi, \sigma : \psi \vee (\phi \wedge \phi \,\mathbf{suf}\, \psi))$.

If a derivation trace is finite or it has an infinite number of progressive steps, we say that the trace is "progressive".

Say a derivation path is "progressive" if it has a progressive derivation trace.

Definition 4.7 (Cyclic Proofs of G3PPL). *A preproof of G3PPL is called a "cyclic proof", if there is a progressive derivation trace over every infinite derivation path.*

Section 5 shows that a cyclic proof concludes a valid conclusion.

In G3PPL, we also say that a sequent is proved if there is a cyclic proof of it. For a labelled PPL formula ϕ, we write G3PPL $\vdash \phi$ (or simply $\vdash \phi$) if sequent $\cdot \Rightarrow \phi$ can be proved in G3PPL.

5 Soundness of G3PPL

We analyze the soundness of G3PPL for any PPL formulas. In G3PPL, a valid PPL formula ϕ can be captured by its equivalent labeled form: $X : \phi$, with $X \in \mathit{TVar}$. Indeed, ϕ is valid if and only if every trace satisfies it: ϕ is valid iff for every trace tr, $tr \models \phi$, iff for every label mapping $\mathfrak{m}$, $\mathfrak{m}(X) \models \phi$, iff for every label mapping $\mathfrak{m}$, $\mathfrak{m} \models X : \phi$. Thus the validity of $X : \phi$ in labeled process logic implies the validity of ϕ in process logic, and vice versa.

Theorem 5.1 (Soundness of G3PPL). *For any labelled PPL formula $X : \phi$ with variable $X \in \mathit{TVar}$, if $\vdash X : \phi$, then $\models X : \phi$.*

We mainly follow the idea behind [6, 21] to prove Theorem 5.1. We proceed by contradiction, assuming that the conclusion of a cyclic proof is invalid. Then by the soundness of the rules of G3PPL there must exist an infinite *invalid derivation path* in which each node is invalid. This violates the definition of the well-foundedness (cf. [8]) itself.

Our proof follows the general strategy of [21], but relaxes the termination condition required there, since the present work operates over a more explicit program model—namely, regular programs. The key technical contribution is the introduction of *minimal counter-example traces* for PPL formulas (Definition 5.5) together with a proof that such traces are always finite with respect to a given PPL formula and a starting trace (Proposition 5.2). This finiteness result is what drives the well-founded descent argument central to the soundness proof.

Below we firstly introduce the concept of well-foundedness and the well-founded relation we rely on, then we focus on the main skeleton of proving Theorem 5.1. Other proof details are given in Appendix A.1.

Definition 5.1 (Well-foundedness). *Given a set S and a partial-order relation $\preceq$ on S, $\preceq$ is called a* well-founded relation *over S, if for any element a in S, there is no infinite descent sequence: $a \succ a_1 \succ a_2 \succ \ldots$ in S. Set S is called a* well-founded *set w.r.t. $\preceq$.*

Definition 5.2 (Relation $\preceq_m$). *Given two finite sets C_1 and C_2 of strings (i.e. a sequence of alphabets), $C_1 \preceq_m C_2$ is defined if either (1) $C_1 = C_2$; or (2) set C_1 can be obtained from C_2 by replacing one or more elements of C_2 each with a finite number of elements, such that for each replaced element str, its replacements $str_1, \ldots, str_n$ ($n \geq 1$) in C_1 are proper suffixes of str.*

Proposition 5.1. *Relation $\preceq_m$ is a well-founded relation.*

We omit the proof of Proposition 5.1. Intuitively, we observe that for two sets C_1 and C_2 such that $C_1 \prec_m C_2$, for each set D_{str} of the strings in C_1 that replace an element str in C_2, the maximum length of the elements of D_{str} is strictly smaller than that of str. By that C_2 is finite, C_1 is finite. And the maximum length of a finite set cannot be infinitely decreasing.

Below we give the main skeleton of the proof of Theorem 5.1, deferring the details of the proof of Lemma 5.1 to Appendix A.1.

Following the main idea above, we first introduce a class of well-founded sets called *minimal counter-example traces of PPL formulas*, which relates to the semantics of PPL formulas along an invalid derivation path. We then state Lemma 5.1, which is the key technical step in the proof of Theorem 5.1 that follows.

The introduction of the trace-associated action strings next is for computing the minimal traces in Definition 5.5.

Definition 5.3 (Trace-associated Action Strings). *Let tr be a trace of a regular program α. The "action string" str associated to tr (w.r.t. α) is a sequence of actions over the transitions of tr.*

We ambiguously use $(tr, str) \in \mathcal{I}(\alpha)$ to denote that tr is a trace of α while str is the associated action string of tr.

For example, let a_1, a_2 be two actions, and let $tr \in \mathcal{I}(a_1 ; a_2)$, then $a_1 a_2$ is the action string of tr. For a trace and a program, the action string of the trace is unique.

Definition 5.4 (Counter-example Traces). *Given a Kripke frame $(\mathcal{S}, \mathcal{I})$ and a dynamic formula ϕ, the set of "counter-example traces" of ϕ beginning with a trace-string pair (tr, str), denoted by $CT(tr, str, \phi)$, is the set of trace-string pairs (that can be used to violate formula ϕ), which is inductively defined as follows:*

1. $CT(tr, str, \psi) =_{df} \{(tr, str)\}$, *if ψ is a non-dynamic formula;*
2. $CT(tr, str, [\langle\alpha\rangle]\psi) =_{df} \{(tr', str') \mid (tr', str') \in \mathcal{I}(\alpha), tr_e = tr'_b\}$, *if ψ is a non-dynamic formula;*
3. $CT(tr, str, [\langle\alpha\rangle]\psi) =_{df} \{(tr' \cdot tr'', str' \cdot str'') \mid (tr', str') \in \mathcal{I}(\alpha), tr_e = tr'_b, (tr'', str'') \in CT(tr \cdot tr', str \cdot str', \psi)\}$, *if ψ is a dynamic formula;*
4. $CT(tr, str, \psi \wedge \varphi) =_{df} CT(tr, str, \psi) \cup CT(tr, str, \varphi)$;
5. $CT(tr, str, \psi \vee \varphi) =_{df} CT(tr, str, \psi) \cap CT(tr, str, \varphi)$.

Note that it is possible that a dynamic formula has some non-dynamical sub-formulas. For example, in a formula $\phi = (p \wedge [\alpha]q)$, ϕ is dynamical, but p is non-dynamical.

Definition 5.5 (Minimum Counter-example Traces). *Given a Kripke frame $(\mathcal{S}, \mathcal{I})$ and a dynamic formula ϕ, the set of "minimum counter-example traces" of ϕ beginning with a trace $tr \in \mathcal{I}^*$, denoted by $MCT(tr, \phi)$, is defined such that*

$$MCT(tr, \phi) =_{df} \{tr' \mid (tr', str) \in CT(tr, \epsilon, \phi), str \text{ is minimal}\},$$

where ϵ is the empty string, satisfying that $w \cdot \epsilon = \epsilon \cdot w = \epsilon$ for any string w; "minimal" means that str is a string with minimal length among all other strings str' such that $(tr', str') \in CT(tr, \epsilon, \phi)$.

The next proposition is crucial to have a well-founded relation $\preceq_m$.

Proposition 5.2. *$MCT(tr, \phi)$ is finite for any trace $tr \in \mathcal{S}^+$ and dynamic formula ϕ.*

The proof of Proposition 5.2 follows from the finite syntactic forms of regular expressions during execution. For any regular program α, the number of its minimum trace-associated action strings is finite. So starting from a trace, there can only be finitely many distinct execution traces whose action strings are minimum.

We call $\mathfrak{m} \in \mathbf{M}$ a *counter-example mapping* of a node v, if it is one of the mappings that make v invalid.

Lemma 5.1. *In a cyclic proof, let $(\sigma : \phi, \sigma' : \phi')$ be a step of a derivation trace over a derivation (v, v') of an invalid derivation path. For any set $MCT(\mathfrak{m}(\sigma), \phi)$ of $\sigma : \phi$ w.r.t a counter-example mappiong $\mathfrak{m}$, there exists a counter-example mapping $\mathfrak{m}'$ and a set $MCT(\mathfrak{m}'(\sigma'), \phi')$ of $\sigma' : \phi'$ such that*

$$MCT(\mathfrak{m}'(\sigma'), \phi') \preceq_m MCT(\mathfrak{m}(\sigma), \phi).$$

Moreover, if $(\sigma : \phi, \sigma' : \phi')$ is a progressive step, then $MCT(\mathfrak{m}'(\sigma'), \phi') \prec_m MCT(\mathfrak{m}(\sigma), \phi)$.

The proof of this lemma is given in Appendix A.1.

Proof of Theorem 5.1. Let $v = (\cdot \Rightarrow X : \phi)$. By contradiction, suppose $\not\models X : \phi$. Then by the soundness of G3PPL, there exists an infinite invalid derivation path P from v (in which every sequent is invalid). Since $\vdash v$ and it forms a cyclic proof, let $\tau_1\tau_2...\tau_k...$ be a progressive trace over P of the form: $v...v_1v_2...v_k...$ $(k \geq 1)$, where each formula τ_i is in v_i $(i \geq 1)$. Let $\tau_i =_{df} \sigma_i : \phi_i$.

Since v_1 is invalid, let $\mathfrak{m}_1$ be one of its counter-example mappings. By Lemma 5.1, from $MCT(\mathfrak{m}_1(\sigma_1), \phi_1)$, there exists an infinite sequence of sets $MCT_1, ..., MCT_k, ...$ $(k \geq 1)$, where each $MCT_i =_{df} MCT(\mathfrak{m}_i(\sigma_i), \phi_i)$ $(i \geq 1)$ with $\mathfrak{m}_i$ a counter-example mapping of node v_i, and which satisfies that $MCT_1 \succeq_m ... \succeq_m MCT_k \succeq_m ....$ Moreover, since trace $\tau_1\tau_2...\tau_k...$ is progressive, there must be an infinite number of $j \geq 1$ such that $MCT_j \succ_m MCT_{j+1}$. This thus forms an infinite descent sequence w.r.t. $\prec_m$, violating the well-foundedness of relation $\preceq_m$ (Proposition 5.1).

□

6 Completeness of G3PPL

We analyze the completeness of G3PPL for any PPL formula ϕ, captured by its equivalent labelled form $X : \phi$ with $X \in TVar$.

Theorem 6.1 (Completeness of G3PPL). *Given a labelled PPL formula $X : \phi$ with $X \in TVar$, if $\models X : \phi$, then $\vdash X : \phi$.*

The proof follows the standard approach for establishing the completeness of labelled proof systems (cf. [6, 9]): we show that G3PPL is capable of deriving every axiom and rule of the proof system of PPL (Table 1). Since the proof system of PPL is complete [12], this suffices to conclude the completeness of G3PPL.

Below we explain how to derive each rule in Table 1. We first establish a Necessitation Lemma (Lemma 6.1) as a generalisation of rule (*Gen*). This lemma is required when deriving the other rules. We then derive the PDL rules, following the proof strategy of [9] adapted to the trace labels of process logic. Finally, we derive the rules specific to PPL, which form the core of the completeness proof for the process-logic part.

Lemma 6.1 (Necessitation). *For any sequent $X : [\alpha]\Gamma \Rightarrow X : [\alpha]\Delta$, where $X : [\alpha]A =_{df} \{X : [\alpha]\phi \mid \phi$ is a PPL formula in $A\}$, there is a derivation from it which is "almost" a cyclic proof, except that it has leaf nodes of the form $X : \Gamma \Rightarrow X : \Delta$. Moreover, there is a progressive trace along each derivation path from the root to a leaf node $X : \Gamma \Rightarrow X : \Delta$.*

In this paper below, we use (*nec*) to represent the derivation of Lemma 6.1.

Proof. We prove by induction on the structure of regular programs.

When α is an atomic program a, without loss of generality, let $X = Y \diamond x$. We have the following derivation:

$$\dfrac{\dfrac{\dfrac{\dfrac{Y \diamond x : \phi \Rightarrow Y \diamond x : \varphi}{Y \diamond y : \phi \Rightarrow Y \diamond y : \varphi}\,(sub)}{Y \diamond x \diamond y : \phi \Rightarrow Y \diamond x \diamond y : \varphi}\,(sub)}{x\,\mathrm{R}_a\,y, Y \diamond x : [a]\phi \Rightarrow Y \diamond x \diamond y : \varphi}\,([a]L)}{Y \diamond x : [a]\phi \Rightarrow Y \diamond x : [a]\varphi}\,([a]R)$$

The derivation above is progressive since rule $([a]R)$ is applied (see Definition 4.6).

When α is a test $\psi?$. Let $X = Y \diamond x$, we have

$$\dfrac{\dfrac{\dfrac{\dfrac{Y \diamond x : \phi \Rightarrow Y \diamond x : \varphi}{Z \diamond x : \phi \Rightarrow Z \diamond x : \varphi}\,(sub)}{Y \diamond x \diamond x : \phi \Rightarrow Y \diamond x \diamond x : \varphi}\,(sub)}{Y \diamond x : [\psi?]\phi, x \diamond x : \psi \Rightarrow Y \diamond x \diamond x : \varphi}\,([\phi?]L)}{Y \diamond x : [\psi?]\phi \Rightarrow Y \diamond x : [\psi?]\varphi}\,([\phi?]R)$$

The derivation is progressive since rule $([\phi?]R)$ is applied.

When α is a sequence program $\alpha_1 ; \alpha_2$, we have

$$\dfrac{\dfrac{\dfrac{X : \phi \Rightarrow X : \varphi}{X : [\alpha_2]\phi \Rightarrow X : [\alpha_2]\varphi}\,IH}{X : [\alpha_1][\alpha_2]\phi \Rightarrow X : [\alpha_1][\alpha_2]\varphi}\,IH}{X : [\alpha_1 ; \alpha_2]\phi \Rightarrow X : [\alpha_1 ; \alpha_2]\varphi}\,([;])\,,$$

where steps *IH* are by induction hypothesis on α_1 and α_2 separately.

When α is a choice program $\alpha_1 \cup \alpha_2$, we have

$$\dfrac{\dfrac{\dfrac{X : \phi \Rightarrow X : \varphi}{X : [\alpha_1]\phi \Rightarrow X : [\alpha_1]\varphi}\,IH}{X : [\alpha_1]\phi, X : [\alpha_2]\phi \Rightarrow X : [\alpha_1]\varphi}\,(wk) \quad \dfrac{\dfrac{X : \phi \Rightarrow X : \varphi}{X : [\alpha_2]\phi \Rightarrow X : [\alpha_2]\varphi}\,IH}{X : [\alpha_1]\phi, X : [\alpha_2]\phi \Rightarrow X : [\alpha_2]\varphi}\,(wk)}{X : [\alpha_1 \cup \alpha_2]\phi \Rightarrow X : [\alpha_1 \cup \alpha_2]\varphi}\,([\cup]))$$

When α is a choice program α_1^*, let $X = Y \diamond x$. We have

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{Y \diamond x : \phi \Rightarrow Y \diamond x : \varphi}{Z \diamond x : \phi \Rightarrow Z \diamond x : \varphi}\,(sub)}{Y \diamond x \diamond x : \phi \Rightarrow Y \diamond x \diamond x : \varphi}\,(sub)}{Y \diamond x \diamond x : \phi, Y \diamond x : [\alpha_1][\alpha_1^*]\phi \Rightarrow Y \diamond x \diamond x : \varphi}\,(wk)}{Y \diamond x : [\mathit{true}?]\phi, Y \diamond x : [\alpha_1][\alpha_1^*]\phi \Rightarrow Y \diamond x : [\mathit{true}?]\varphi}\,([\phi?]L,[\phi?]R)}{Y \diamond x : [\alpha_1^*]\phi \Rightarrow Y \diamond x : [\mathit{true}?]\varphi}\,([*]) \qquad \text{Cont.}}{Y \diamond x : [\alpha_1^*]\phi \Rightarrow Y \diamond x : [\alpha_1^*]\varphi}\,([*])\,,$$

$$\dfrac{\dfrac{\dfrac{Y \diamond x : [\alpha_1^*]\phi \Rightarrow Y \diamond x : [\alpha_1^*]\varphi\ (\text{bud})}{Y \diamond x : [\alpha_1][\alpha_1^*]\phi \Rightarrow Y \diamond x : [\alpha_1][\alpha_1^*]\varphi}\,IH}{Y \diamond x : [\mathit{true}?]\phi, Y \diamond x : [\alpha_1][\alpha_1^*]\phi \Rightarrow Y \diamond x : [\alpha_1][\alpha_1^*]\varphi}\,(wk)}{\text{Cont.} : Y \diamond x : [\alpha_1^*]\phi \Rightarrow Y \diamond x : [\alpha_1][\alpha_1^*]\varphi}\,[*]\,,$$

where the left derivation branch is progressive since rule $[\phi?]R$ is applied; on the right derivation branch (from "Cont."), by induction-hypothesis step *IH*, we know that the derivation path from the root node to its bud is progressive.

□

The Necessitation lemma is in fact a general form of the rule (Gen). Thus from it we can trivially derive (Gen). For the other rules of PDL shown in Table 1, the derivations of the rules (3) - (6) follow directly by applying the rules $([\cup])$, $([;])$, $([\phi?])$ and $([*])$ of G3PPL. The derivation of rule (MP) is a special case of rule (cut) of G3PPL. Below, we only show the derivations for the rules (1), (2), (7) in G3PPL.

The derivation for formula $X : [\alpha](\phi \to \varphi) \to ([\alpha]\phi \to [\alpha]\varphi)$ (corresponding to rule (1)) with $X \in TVar$ is shown as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{X:\phi \Rightarrow X:\phi, X:\varphi}\,(ax) \quad \dfrac{}{X:\varphi, X:\phi \Rightarrow X:\varphi}\,(ax)}{X:(\phi\to\varphi), X:\phi \Rightarrow X:\varphi}\,(\to L)}{X:[\alpha](\phi\to\varphi), X:[\alpha]\phi \Rightarrow X:[\alpha]\varphi}\,nec}{X:[\alpha](\phi\to\varphi) \Rightarrow X:([\alpha]\phi \to [\alpha]\varphi)}\,(\to R)}{\cdot \Rightarrow X:[\alpha](\phi\to\varphi) \to ([\alpha]\phi \to [\alpha]\varphi)}\,(\to R)$$

The derivation for formula $X : [\alpha](\phi \wedge \varphi) \leftrightarrow ([\alpha]\phi \wedge [\alpha]\varphi)$ with $X \in TVar$ (corresponding to rule (2)) is shown as follows for both directions separately:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{X:\phi, X:\varphi \Rightarrow X:\phi}\,(ax)}{X:(\phi\wedge\varphi) \Rightarrow X:\phi}\,(\wedge L)}{X:[\alpha](\phi\wedge\varphi) \Rightarrow X:[\alpha]\phi}\,nec \quad \dfrac{\dfrac{\dfrac{}{X:\phi, X:\varphi \Rightarrow X:\varphi}\,(ax)}{X:(\phi\wedge\varphi) \Rightarrow X:\varphi}\,(\wedge L)}{X:[\alpha](\phi\wedge\varphi) \Rightarrow X:[\alpha]\varphi}\,nec}{X:[\alpha](\phi\wedge\varphi) \Rightarrow X:([\alpha]\phi \wedge [\alpha]\varphi)}\,(\wedge R)}{\cdot \Rightarrow X:[\alpha](\phi\wedge\varphi) \to ([\alpha]\phi \wedge [\alpha]\varphi)}\,(\to R)$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{X:\phi, X:\varphi \Rightarrow X:\phi}\,(ax) \quad \dfrac{}{X:\phi, X:\varphi \Rightarrow X:\varphi}\,(ax)}{X:\phi, X:\varphi \Rightarrow X:(\phi\wedge\varphi)}\,(\wedge R)}{X:[\alpha]\phi, X:[\alpha]\varphi \Rightarrow X:[\alpha](\phi\wedge\varphi)}\,nec}{X:([\alpha]\phi \wedge [\alpha]\varphi) \Rightarrow X:[\alpha](\phi\wedge\varphi)}\,(\wedge L)}{\cdot \Rightarrow X:([\alpha]\phi \wedge [\alpha]\varphi) \to [\alpha](\phi\wedge\varphi)}\,(\to R)$$

The derivation for formula $X : \phi \wedge [\alpha^*](\phi \to [\alpha]\phi) \to [\alpha^*]\phi$ with $X \in TVar$ (Corresponding to rule (7)) is shown as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\text{Lemma 6.2}}{X:\phi, X:[\alpha^*](\phi\to[\alpha]\phi) \Rightarrow X:[true?]\phi}\quad \text{Cont.}}{X:\phi, X:[\alpha^*](\phi\to[\alpha]\phi) \Rightarrow X:[\alpha^*]\phi\ (\text{companion})}\,([*])}{X:\phi \wedge [\alpha^*](\phi\to[\alpha]\phi) \Rightarrow X:[\alpha^*]\phi}\,(\wedge L)}{\cdot \Rightarrow X:\phi \wedge [\alpha^*](\phi\to[\alpha]\phi) \to [\alpha^*]\phi}\,(\to R)$$

,

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{X:\phi, X:[\alpha][\alpha^*](\phi\to[\alpha]\phi) \Rightarrow X:[\alpha][\alpha^*]\phi, X:\phi}\,(ax) \quad \text{Cont. 2}}{X:\phi, X:(\phi\to[\alpha]\phi), X:[\alpha][\alpha^*](\phi\to[\alpha]\phi) \Rightarrow X:[\alpha][\alpha^*]\phi}\,(\to L)}{X:\phi, X:[true?](\phi\to[\alpha]\phi), X:[\alpha][\alpha^*](\phi\to[\alpha]\phi) \Rightarrow X:[\alpha][\alpha^*]\phi}\,lem}{\text{Cont.} : X:\phi, X:[\alpha^*](\phi\to[\alpha]\phi) \Rightarrow X:[\alpha][\alpha^*]\phi}\,[*]$$

,

$$\dfrac{\dfrac{X:\phi, X:[\alpha^*](\phi\to[\alpha]\phi) \Rightarrow X:[\alpha^*]\phi\ (\text{bud})}{X:[\alpha]\phi, X:[\alpha][\alpha^*](\phi\to[\alpha]\phi) \Rightarrow X:[\alpha][\alpha^*]\phi}\,nec}{\text{Cont. 2} : X:\phi, X:[\alpha]\phi, X:[\alpha][\alpha^*](\phi\to[\alpha]\phi) \Rightarrow X:[\alpha][\alpha^*]\phi}\,(wk)$$

The above derivation also relies on Lemma 6.2 as follows

Lemma 6.2. *For any $X \in LVar$ and formula ϕ, sequents $X : \phi \Rightarrow X : [true?]\phi$ and $X : [true?]\phi \Rightarrow X : \phi$ are provable in G3PPL.*

The proof of Lemma 6.2 is given in Appendix 6. The derivation step tagged by "*lem*" according to Lemma 6.2. It is a shorthand of firstly applying (cut) (with the additional formula $X : (\phi \rightarrow [\alpha]\phi)$) and then applying $X : [true?]\phi \Rightarrow X : \phi$ in Lemma 6.2. The derivation path from node "companion" to node "bud" is progressive due to that a Necessitation lemma (*nec*) is applied.

It remains to prove the special rules (i)–(xv) for PPL in Table 1. As a representative, below we give the derivation for one direction of rule (iii). Refer to Appendix 6 for the derivation of the other direction and the derivations of the other rules.

For one direction of rule (iii), we need to prove

$$(iii.a)\ X : (\phi\ \mathbf{suf}\ \psi) \vee (\phi\ \mathbf{suf}\ \chi) \Rightarrow X : \phi\ \mathbf{suf}(\psi \vee \chi).$$

Without loss of generality, let $X = x \diamond Y$. By $(\vee L)$, we need

$(iii.a1)$ $x \diamond Y : (\phi\ \mathbf{suf}\ \psi) \Rightarrow x \diamond Y : \phi\ \mathbf{suf}(\psi \vee \chi)$.
$(iii.a2)$ $x \diamond Y : (\phi\ \mathbf{suf}\ \chi) \Rightarrow x \diamond Y : \phi\ \mathbf{suf}(\psi \vee \chi)$.

For (iii.a1), by applying $(\mathbf{suf})$ on both left and right sides, we have

$$Y : \psi \vee (\phi \wedge \phi\ \mathbf{suf}\ \psi) \Rightarrow Y : \psi \vee \chi, Y : \phi \wedge \phi\ \mathbf{suf}(\psi \vee \chi).$$

Applying rule $(\vee L)$, we have

$(iii.a11)$ $Y : \psi \Rightarrow Y : \psi \vee \chi, Y : \phi \wedge \phi\ \mathbf{suf}(\psi \vee \chi)$, which can be proved by $(\vee R)$ and (ax).
$(iii.a12)$ $Y : (\phi \wedge \phi\ \mathbf{suf}\ \psi) \Rightarrow Y : \psi \vee \chi, Y : \phi \wedge \phi\ \mathbf{suf}(\psi \vee \chi)$, from which, by splitting $Y : \phi \wedge \phi\ \mathbf{suf}(\psi \vee \chi)$ with rule $(\wedge R)$, we need to prove $Y : (\phi \wedge \phi\ \mathbf{suf}\ \psi) \Rightarrow Y : \psi \vee \chi, Y : \phi$, which can be closed by (ax), and

$$(iii.a121)\ Y : (\phi \wedge \phi\ \mathbf{suf}\ \psi) \Rightarrow Y : \psi \vee \chi, Y : \phi\ \mathbf{suf}(\psi \vee \chi).$$

By (wk), we have

$$(iii.a1211)\ Y : \phi\ \mathbf{suf}\ \psi \Rightarrow Y : \phi\ \mathbf{suf}(\psi \vee \chi),$$

which is exactly (iii.a1) (by replacing Y with X). The derivation path from (iii.a1) to (iii.a1211) is progressive since during the process $(\mathbf{suf})$ is applied.

Case for (iii.a2) is quite similar. By applying $(\mathbf{suf})$ on both left and right sides, there are

$(iii.a21)$ $Y : \chi \Rightarrow Y : \psi \vee \chi, Y : \phi \wedge \phi\ \mathbf{suf}(\psi \vee \chi)$, which can be proved by $(\vee R)$ and (ax).
$(iii.a22)$ $Y : (\phi \wedge \phi\ \mathbf{suf}\ \chi) \Rightarrow Y : \psi \vee \chi, Y : \phi \wedge \phi\ \mathbf{suf}(\psi \vee \chi)$, from which, by splitting $Y : \phi \wedge \phi\ \mathbf{suf}(\psi \vee \chi)$ with rule $(\wedge R)$, we need to prove $Y : (\phi \wedge \phi\ \mathbf{suf}\ \chi) \Rightarrow Y : \psi \vee \chi, Y : \phi$, which can be closed by (ax), and

$$(iii.a221)\ Y : (\phi \wedge \phi\ \mathbf{suf}\ \chi) \Rightarrow Y : \psi \vee \chi, Y : \phi\ \mathbf{suf}(\psi \vee \chi).$$

By (wk), we have

$$(iii.a2211)\ Y : \phi\ \mathbf{suf}\ \chi \Rightarrow Y : \phi\ \mathbf{suf}(\psi \vee \chi),$$

which is exactly (iii.a2) (by replacing Y with X). The derivation path from (iii.a2) to (iii.a2211) is progressive since during the process $(\mathbf{suf})$ is applied.

7 Labelled First-Order Process Logic

At the first-ordered level of process logic, assignments generate program-state information that must be tracked explicitly throughout derivations. We address this by replacing the trace world-variables of PPL labels with *program updates*, which record the cumulative effect of assignments as structured label components. The following subsections define update-based labels and labelled FOPL formulas, present the proof system G3FOPL, and analyse its soundness and relative completeness.

7.1 Labelled FOPL Formulas

Program updates, introduced in [2] as a first-order label structure for capturing program configurations during derivations, provide the foundation for the labelled treatment of FOPL. An update $\mathcal{U}$ records the accumulated effect of assignment steps as an explicit substitution applied to the current world variable, making the program configuration available inside the proof system without requiring rule-specific axioms for each assignment form.

Definition 7.1 (Program Updates). *A program update $\mathcal{U}$ in FOPL is defined as a first-ordered configuration as follows:*

$$\mathcal{U} =_{df} x \mid \{u := e\}\mathcal{U},$$

where $x \in WVar$, $u \in Var$ and $e \in \mathfrak{A}$.

We use $\mathbf{U}$ to represent the set of all updates in FOPL. $\{u := e\}$ captures the effect of assigning the value of e to the variable u in the current context.

The labels of FOPL enrich those of PPL with the explicit structures of program updates $\mathcal{U}$. We give the definitions of labels and label mappings of FOPL in the following definitions.

Definition 7.2 (Labels of FOPL). *A label σ in FOPL is a trace defined as follows:*

$$\sigma =_{df} \varepsilon \mid \mathcal{U} \mid X \mid \sigma \diamond \sigma,$$

where $X \in TVar$, ε is the empty label, $\diamond$ is the concatenation operator; $\mathcal{U}$ is a program update.

We use $\mathbf{L}_{fo}$ to denote the set of all labels of FOPL.

Definition 7.3 (Label mappings of FOPL). *A world mapping $\mathfrak{w}$ in FOPL is a function from $\mathbf{U}$ to $\mathcal{S}_{fo}$, defined inductively as follows:*

1. *$\mathfrak{w}$ maps a variable $x \in WVar$ to a world in $\mathcal{S}_{fo}$.*
2. *$\mathfrak{w}(\{u := e\}\mathcal{U}) =_{df} \mathfrak{m}(\mathcal{U})[u \mapsto \mathfrak{w}(\mathcal{U})(e)]$.*

A label mapping $\mathfrak{m}$ is a function from $\mathbf{L}_{fo}$ to $\mathcal{S}^*_{fo}$, defined as follows:

i *$\mathfrak{m}$ maps a variable $X \in TVar$ to a trace of worlds in $\mathcal{S}^*_{fo}$.*
ii *$\mathfrak{m}(\varepsilon) =_{df} \epsilon$.*
iii *$\mathfrak{m}(\mathcal{U}) =_{df} \mathfrak{w}(\mathcal{U})$.*
iv *$\mathfrak{m}(\sigma_1 \diamond \sigma_2) =_{df} \mathfrak{m}(\sigma_1) \cdot \mathfrak{m}(\sigma_2)$.*

We use $\mathbf{M}_{fo}$ to denote the set of all label mappings from $\mathbf{L}_{fo}$ to $\mathcal{S}^*_{fo}$.

A *labeled FOPL formula* is just defined as in PPL (Definition 4.3) (but replacing $\mathbf{L}$ as $\mathbf{L}_{fo}$).

The semantics of labelled FOPL formulas is introduced as follows.

Rule	Schematic form
Inherited rules	All rules of G3PPL except $([a]R)$ and $([a]L)$, with labels interpreted in $\mathbf{L}_{fo}$
$([u := e])$	$\dfrac{\sigma \diamond \mathcal{U} \diamond \{u := e\}\mathcal{U} : \phi}{\sigma \diamond \mathcal{U} : [u := e]\phi}\ ([u:=e])$
(at)	$\dfrac{\mathcal{U} : e_1 \leq e_2}{\mathcal{U} \diamond \sigma : e_1 \leq e_2}\ (at)$
(ter)	$\dfrac{}{\Gamma \Rightarrow \Delta}\ (ter)$, where each formula of $\Gamma \Rightarrow \Delta$ is of the form $\mathcal{U} : e_1 \leq e_2$
$(\forall R)$	$\dfrac{\Gamma \Rightarrow \phi[v/u], \Delta}{\Gamma \Rightarrow \forall u.\phi, \Delta}\ (\forall R)$, where $v \in \mathit{Var}$ is fresh w.r.t. Γ, Δ, ϕ
$(\forall L)$	$\dfrac{\Gamma, \phi[e/u] \Rightarrow \Delta}{\Gamma, \forall u.\phi \Rightarrow \Delta}\ (\forall L)$, where e is a term

Table 6. Rules specific to labelled first-order process logic.

Definition 7.4 (Semantics of Labelled FOPL Formulas). *Given the Kripke frame $(\mathcal{S}_{fo}, \mathcal{I}_{fo})$ of FOPL and the set $\mathbf{M}_{fo}$ of label mappings, the satisfaction relation $\mathfrak{m} \models_{\mathbf{M}_{fo}} \sigma : \phi$ of a labeled formula $\sigma : \phi$ by a label mapping $\mathfrak{m}$ (w.r.t. $\mathbf{M}_{fo}$) is defined such that*

$$\mathfrak{m} \models_{\mathbf{M}_{fo}} \sigma : \phi, \ \textit{if}\ \mathfrak{m}(\sigma) \models_{(\mathcal{S}_{fo}, \mathcal{I}_{fo})} \phi.$$

7.2 System G3FOPL

The proof system G3FOPL for labelled FOPL formulas extends G3PPL by specializing the rules for atomic programs and first-order arithmetic formulas, while keeping all other rules unchanged. As shown in Table 6, the rule $([u := e])$ replaces the pair $([a]R)$ and $([a]L)$ in G3PPL: it appends the update $\{u := e\}$ to the label, recording the assignment effect in the label structure rather than substituting into the formula. The rule (at) replaces rule (p) for the evaluation of arithmetic predicates at an update: it strips the trailing label σ and evaluates $e_1 \leq e_2$ at the update $\mathcal{U}$ alone. The rule (ter) closes a derivation branch when all remaining formulas are arithmetic sequents of the form $\mathcal{U} : e_1 \leq e_2$; such sequents can be discharged by an external arithmetic decision procedure or SMT solver. The rules $(\forall R)$ and $(\forall L)$ handle universal quantification in the standard way as in FODL.

All other rules of G3FOPL are inherited from G3PPL with labels interpreted in $\mathbf{L}_{fo}$, and with each world variable x in those rules replaced by a program update $\mathcal{U} \in \mathbf{U}$. For example, rule $(\mathbf{suf})$ in G3FOPL takes the form

$$\frac{\sigma : \psi \vee (\phi \wedge \phi\ \mathbf{suf}\ \psi)}{\mathcal{U} \diamond \sigma : \phi\ \mathbf{suf}\ \psi}\ (\mathbf{suf}),$$

with $\sigma \in \mathbf{L}_{fo}$, following the same trace-reduction principle as in G3PPL.

The soundness of the G3FOPL rules follows directly from the semantics of labelled FOPL formulas.

Theorem 7.1. *Each rule of G3FOPL is sound.*

The cyclic proof structure of G3FOPL can be defined the same as that of G3PPL, except that for its progressive derivation traces (Definition 4.6), a progressive step can no longer be an instance of rule $([a]R)$, instead can be their

replacements in G3FOPL, i.e., rule ($[u := e]$) for the right side:

$$\frac{\Gamma \Rightarrow \sigma \diamond \mathcal{U} : [u := e]\phi}{\Gamma \Rightarrow \sigma \diamond \mathcal{U} \diamond \{u := e\}\mathcal{U} : \phi} \ ([u:=e]R).$$

Similarly, in G3FOPL, we say that a sequent is proved also when there is a cyclic proof of it. For a labelled FOPL formula ϕ, we write G3FOPL $\vdash \phi$ (or simply $\vdash \phi$) if sequent $\cdot \Rightarrow \phi$ can be proved in G3FOPL.

7.3 Soundness and Relative Completeness of G3FOPL

We discuss about the soundness and completeness of the proof system G3FOPL w.r.t all FOPL formulas. The soundness proof for G3FOPL follows the same structure as that of G3PPL, adapting the cases for the rules ($[u := e]$), ($\forall R$), and ($\forall L$) that are specific to the first-order setting. The completeness of G3FOPL is obtained by adapting the standard completeness argument for FODL [12] to the trace semantics of FOPL.

Theorem 7.2 (Soundness of G3FOPL). *For any labelled FOPL formula $X : \phi$, where $X \in TVar$, if $\vdash X : \phi$, then $\models X : \phi$.*

The proof of Theorem 7.2 proceeds along the same lines as the proof of soundness for G3PPL. The argument is modified in two respects. First, the case for progressive steps corresponding to rule ($[a]R$) in G3PPL is replaced by the case for rule ($[u := e]R$) in G3FOPL; the proof carries through essentially unchanged, since the update rule records the effect of the assignment in the label and the same multiset-ordering argument applies. Second, the cases for rules ($\forall R$) and ($\forall L$) must be handled; these are straightforward and proceed similarly to the case for rule (sub) in G3PPL, as the quantifier rules introduce or eliminate a label variable in a way that respects the counter-example mappings.

Theorem 7.3 (Completeness of G3FOPL). *For a labelled FOPL formula $X : \phi$ with $X \in TVar$ a label variable, if $\models X : \phi$, then $\vdash X : \phi$.*

The completeness of G3FOPL follows the main approach used for the completeness of FODL [10, 12]. The argument proceeds in two steps. We first establish two auxiliary lemmas—Lemma 7.1 and Lemma 7.2—that are required for the main completeness argument. The full proof of Theorem 7.3 then follows from these two lemmas and the structure of FOPL formulas.

Lemma 7.1. *For any FOPL formula ϕ, there exists a pure arithmetical FOL formula $\psi^\flat$ such that $\models \psi^\flat \leftrightarrow \phi$.*

We usually write a formula ϕ as $\phi^\flat$ to address that it is a pure arithmetical FOL formula.

We omit the proof of Lemma 7.1. Here is an informal explanation. It is well known that temporal properties can be encoded in first-order form over suitable trace positions (cf. [19]). On the other hand, any FODL formula can be expressed by a pure arithmetical FOL formula. The result of Lemma 7.1 is direct by combining these two facts.

Lemma 7.2. *For any FOPL formulas ϕ and ψ and label variable X, let $op \in \{[\alpha], \langle\alpha\rangle\}$ for any program α, if $\models X : \phi^\flat \Rightarrow X : op\,\psi^\flat$, then $\vdash X : \phi^\flat \Rightarrow X : op\,\psi^\flat$.*

The proof of this lemma is given in Appendix A.3.

Proof of Theorem 7.3. For a labelled formula $X : \phi$, ϕ is semantically equivalent to a conjunctive normal form: $C_1 \wedge ... \wedge C_n$ ($n \geq 1$). Each clause C_i ($1 \leq i \leq n$) is a disjunction of literals: $C_i = l_{i,1} \vee ... \vee l_{i,m_i}$, where $l_{i,j}$ ($1 \leq i \leq n, 1 \leq j \leq m_i$) is an atomic FODL formula or its negation. By the rule for FOL formulas (Table 4 plus the rules ($\forall R$) and ($\forall L$)), to prove formula $X : \phi$, it is enough to show that for each clause C_i, $\models X : C_i$ implies $\vdash X : C_i$. We prove by induction on the sum n of the appearances of modalities $[\cdot]$ and $\langle\cdot\rangle$ in C_i.

If $n = 0$, there are no appearances of $[\alpha]$ or $\langle\alpha\rangle$ in C_i, so C_i is an FOL formula. By rule (*Ter*), immediately we have $\vdash X : C_i$.

If $n > 0$, without loss of generality, let $C_i = \psi_1 \vee op\,\psi_2$ where $op \in \{[\alpha], \langle\alpha\rangle\}$. Now we prove $\vdash X : (\psi_1 \vee op\,\psi_2)$, which is equivalent to prove

$$\vdash X : \neg\psi_1 \Rightarrow X : op\,\psi_2$$

according to $(\vee R)$ and $(\neg R)$. By Lemma 7.1, there are pure arithmetical FOL formulas $\phi_1^\flat$ and $\phi_2^\flat$ such that $\models \phi_1^\flat \leftrightarrow \neg\psi_1$ and $\models \phi_2^\flat \leftrightarrow \psi_2$. Therefore, we have both $\models X : \neg\psi_1 \Rightarrow X : op\,\phi_2^\flat$ and $\models (X : \phi_2^\flat \Rightarrow X : \psi_2)$. By assumption, we have both

$$\vdash X : \neg\psi_1 \Rightarrow X : op\,\phi_2^\flat \tag{1}$$

and

$$\vdash X : \phi_2^\flat \Rightarrow X : \psi_2. \tag{2}$$

From (2), by Necessitation Lemma, we have

$$\vdash X : op\,\phi_2^\flat \Rightarrow X : op\,\psi_2. \tag{3}$$

On the other hand, from (1), by Lemma 7.2, we have

$$\vdash X : \neg\psi_1 \Rightarrow X : op\,\phi_2^\flat. \tag{4}$$

The result is directly obtained by (3) and (4).

□

8 Conclusion

This paper has developed a labelled cyclic proof-theoretic framework for process logic, encompassing both the propositional level (G3PPL) and the first-ordered level (G3FOPL). The central contribution is a labelling discipline that makes trace-execution information explicit in derivations: propositional labels record traces over atomic actions, while first-order labels record sequences of program updates. By making this information part of the label structure rather than encoding it into the formula, both systems admit uniform, modular inference rules for the trace connectives of process logic and the modalities of first-order dynamic logic.

The theoretic results establish that the labelled framework is both sound and complete. Soundness follows from combining local soundness of the individual rules with a global cyclic condition: every infinite derivation path that satisfies the progress condition gives rise to an infinite descent in the well-founded multiset ordering on counter-example traces, which is a contradiction. Completeness is established separately for each system: G3PPL is proved complete by showing that it can derive every axiom and rule of the Hilbert-style proof system for PPL; while the completeness of G3FOPL is proved by adapting the classical completeness argument for FODL to the trace-based setting of process logic.

Together, these results show that labelling provides a principled and uniform mechanism for cyclic reasoning about trace-based program properties.

Directions for future work include the application of G3FOPL to pratical programs or languages, the development of automated proof-search strategies and the mechanization for G3PPL and G3FOPL, and the extension of the framework to richer program models such as concurrent or probabilistic processes.

References

[1] Hammad Ahmad and Jean-Baptiste Jeannin. 2021. A Program Logic to Verify Signal Temporal Logic Specifications of Hybrid Systems. In *Proceedings of the 24th ACM International Conference on Hybrid Systems: Computation and Control (HSCC 2021)*. ACM, Nashville, TN, USA, 1–11. https://doi.org/10.1145/3447928.3456648

[2] Bernhard Beckert and Daniel Bruns. 2013. Dynamic Logic with Trace Semantics. In *CADE 2013*. Springer Berlin Heidelberg, 315–329.

[3] Bernhard Beckert, Vladimir Klebanov, and Benjamin Weiß. 2016. *Dynamic Logic for Java*. Springer International Publishing, 49–106.

[4] Bernhard Beckert and Steffen Schlager. 2001. A Sequent Calculus for First-Order Dynamic Logic with Trace Modalities. In *Automated Reasoning*. Springer Berlin Heidelberg, Berlin, Heidelberg, 626–641.

[5] Gérard Berry and Georges Gonthier. 1992. The Esterel synchronous programming language: design, semantics, implementation. *Science of Computer Programming* 19, 2 (1992), 87 – 152.

[6] James Brotherston, Richard Bornat, and Cristiano Calcagno. 2008. Cyclic proofs of program termination in separation logic. *SIGPLAN Not.* 43, 1 (2008), 101–112.

[7] James Brotherston and Alex Simpson. 2007. Complete Sequent Calculi for Induction and Infinite Descent. In *LICS 2007*. 51–62.

[8] Nachum Dershowitz and Zohar Manna. 1979. Proving termination with multiset orderings. In *Automata, Languages and Programming*. Springer Berlin Heidelberg, Berlin, Heidelberg, 188–202.

[9] Simon Docherty and Reuben N. S. Rowe. 2019. A Non-wellfounded, Labelled Proof System for Propositional Dynamic Logic. In *TABLEAUX 2019*. Springer International Publishing, 335–352.

[10] David Harel. 1979. *First-Order Dynamic Logic*. Lecture Notes in Computer Science (LNCS), Vol. 68. Springer.

[11] David Harel, Dexter Kozen, and Rohit Parikh. 1982. Process Logic: Expressiveness, Decidability, Completeness. *J. Comput. System Sci.* 25, 2 (1982), 144–170.

[12] David Harel, Dexter Kozen, and Jerzy Tiuryn. 2000. *Dynamic Logic*. MIT Press.

[13] Jean-Baptiste Jeannin and André Platzer. 2014. dTL2: Differential Temporal Dynamic Logic with Nested Temporalities for Hybrid Systems. In *Automated Reasoning*. Springer International Publishing, Cham, 292–306.

[14] R. Milner. 1982. *A Calculus of Communicating Systems*. Springer-Verlag, Berlin, Heidelberg.

[15] Sara Negri. 2005. Proof Analysis in Modal Logic. *Journal of Philosophical Logic* 34, 5–6 (2005), 507–544. https://doi.org/10.1007/s10992-005-2267-3

[16] André Platzer. 2007. Differential Dynamic Logic for Verifying Parametric Hybrid Systems.. In *TABLEAUX 2007* (2007-09-20). Springer Berlin Heidelberg, 216–232.

[17] André Platzer. 2007. A Temporal Dynamic Logic for Verifying Hybrid System Invariants. In *Logical Foundations of Computer Science*. Springer Berlin Heidelberg, Berlin, Heidelberg, 457–471.

[18] André Platzer. 2018. *Logical Foundations of Cyber-Physical Systems*. Springer.

[19] Amir Pnueli. 1977. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science (SFCS 1977)*. 46–57. https://doi.org/10.1109/SFCS.1977.32

[20] Colin Stirling and David Walker. 1991. Local model checking in the modal mu-calculus. *Theor. Comput. Sci.* 89, 1 (1991), 161–177.

[21] Yuanrui Zhang. 2025. Parameterized Dynamic Logic – Towards A Cyclic Logical Framework for General Program Specification and Verification. arXiv:2404.18098 [cs.LO] https://arxiv.org/abs/2404.18098

[22] Yuanrui Zhang and Zhiming Liu. 2024. A dynamic logic with branching modalities. *Journal of Logical and Algebraic Methods in Programming* 136 (2024), 100921. https://doi.org/10.1016/j.jlamp.2023.100921

[23] Yuanrui Zhang and Zhibin Yang. 2026. A Generic Dynamic Logic for Program Reasoning Based on Operational Semantics. In *Dependable Software Engineering. Theories, Tools, and Applications*, Amir Goharshady and Christoph Haase (Eds.). Springer Nature Singapore, Singapore, 93–112.

A Other Proofs of Main Theories

A.1 Soundness Proofs for G3PPL

Content of Lemma 7.2: In a cyclic proof, let $(\sigma : \phi, \sigma' : \phi')$ be a step of a derivation trace over a derivation (ν, ν') of an invalid derivation path. For any set $MCT(\mathfrak{m}(\sigma), \phi)$ of $\sigma : \phi$ w.r.t a counter-example mappiong $\mathfrak{m}$, there exists a counter-example mapping $\mathfrak{m}'$ and a set $MCT(\mathfrak{m}'(\sigma'), \phi')$ of $\sigma' : \phi'$ such that

$$MCT(\mathfrak{m}'(\sigma'), \phi') \preceq_m MCT(\mathfrak{m}(\sigma), \phi).$$

Moreover, if $(\sigma : \phi, \sigma' : \phi')$ is a progressive step, then $MCT(\mathfrak{m}'(\sigma'), \phi') \prec_m MCT(\mathfrak{m}(\sigma), \phi)$.

Proof of Lemma 7.2. It is enough to consider the cases when the derivation step (ν, ν') is an instance of rule $([a]R)$, $([a]L)$, $([\phi?]R)$, $([\phi?]L)$, (**suf**) and (*sub*). Other cases are trivial.

Case for rule $([a]R)$**:** If from node ν rule $([a]R)$ is applied with $\tau =_{df} \sigma \diamond x : [a]\phi$ the target formula, let $\tau' = (\sigma \diamond x \diamond y : \phi)$ for some y, so $\nu = (\Gamma \Rightarrow \tau, \Delta)$ and $\nu' = (\Gamma, xR_a y \Rightarrow \tau', \Delta)$. In this case, (ν, ν') is a progressive step (Definition 4.6). Since $\mathfrak{m}$ is a counter-example of ν, $\mathfrak{m} \not\models \tau$, so $MCT(\mathfrak{m}(\sigma \diamond x), [a]\phi) \neq \emptyset$. By the soundness of rule $([a]R)$, for each trace $s_1 ... s_n \in MCT(\mathfrak{m}(\sigma \diamond x \diamond y), \phi)$ $(n \geq 0)$, trace $ss_1 ... s_n \in MCT(\mathfrak{m}(\sigma \diamond x), [a]\phi)$ with $ss_1 \in \mathcal{I}(a)$ has $a_1 ... a_n$ as its proper suffix. On the other hand, by Proposition 5.2, $MCT(\mathfrak{m}(\sigma \diamond x), [a]\phi)$ and $MCT(\mathfrak{m}(\sigma \diamond x \diamond y), \phi)$ are also finite. Therefore $MCT(\mathfrak{m}(\sigma \diamond x \diamond y), \phi) \prec_m MCT(\mathfrak{m}(\sigma \diamond x), [a]\phi)$ by Definition 5.2.

Case for rule $([a]L)$**:** Similar to the case $([a]R)$, except that it is possible that $MCT(\mathfrak{m}(\sigma \diamond x), [a]\phi) = \emptyset$ (when there is no trace of $MCT(\mathfrak{m}(\sigma \diamond x), [a]\phi)$, $\mathfrak{m}$ can also be a counter-example mapping for ν too). So, we have $MCT(\mathfrak{m}(\sigma \diamond x \diamond y), \phi) \preceq_m MCT(\mathfrak{m}(\sigma \diamond x), [a]\phi)$. (Recall that in this case, (ν, ν') is not a progressive step.)

Case for rule $([\phi?]R)$**:** Similar to the case of $([a]R)$ and we omit it.

Case for rule $([\phi?]L)$**:** Similar to the case of $([a]L)$ and we omit it.

Case for rule (***sub***)**:** If from node ν a substitution rule (*sub*) is applied, let $\tau = \sigma[\sigma'/U] : \phi$ be the target formula of ν, where σ' is a label, $U \in LVar$ is a label variable. Then $\tau' = \sigma : \phi$. Let $\mathfrak{m}'$ be the label mapping such that $\mathfrak{m}'(V) =_{df} \mathfrak{m}(\sigma')$; $\mathfrak{m}'(U) =_{df} \mathfrak{m}(U)$ for any other $U \in LVar$. So, we have $\mathfrak{m}'(\sigma) = \mathfrak{m}(\sigma[\sigma'/V])$. Hence $MCT(\mathfrak{m}(\sigma[\sigma'/V])), \phi) = MCT(\mathfrak{m}'(\sigma), \phi)$.

Case for rule (**suf**)**:** If from node ν rule (**suf**) is applied, let $\tau = x \diamond \sigma : (\phi \,\mathbf{suf}\, \psi)$ be the target formula of ν, then $\tau' = \sigma : \psi \vee (\phi \wedge \phi \,\mathbf{suf}\, \psi)$. By the soundness of rule (**suf**) and the definition of counter-example traces (Definition 5.4), we have $MCT(\mathfrak{m}(x \diamond \sigma), \phi \,\mathbf{suf}\, \psi) = \{\mathfrak{m}(x)\mathfrak{m}(\sigma)\}$ and $MCT(\mathfrak{m}(\sigma), \psi \vee (\phi \wedge \phi \,\mathbf{suf}\, \psi)) = \{\mathfrak{m}(\sigma)\}$. Since $\mathfrak{m}(\sigma)$ is a proper suffix of $\mathfrak{m}(x)\mathfrak{m}(\sigma)$, we obtain the result. □

A.2 Completeness Proofs of G3PPL

Content of Lemma 6.2: For any $X \in LVar$ and formula ϕ, sequents $X : \phi \Rightarrow X : [true?]\phi$ and $X : [true?]\phi \Rightarrow X : \phi$ are provable in G3PPL.

Lemma 6.2 is subsequently used in the proof of Lemma A.2.

Proof of Lemma 6.2. We only prove $X : \phi \Rightarrow X : [true?]\phi$. The other direction is similar.

Without loss of generality, let $X = Y \diamond x$. By $([\phi?]R)$ and (wk), we need

$$(A2) \quad Y \diamond x : \phi \Rightarrow Y \diamond x \diamond x : \phi$$

We prove $(A2)$ by induction on the structure of formula ϕ.

The base step is when ϕ is an atomic formula. Then $(A2)$ can be trivially derived by applying (p) and (ax).

For the inductive step, consider then $\phi = \mathbf{f}\,\psi$ and $\phi = \phi_1\ \mathbf{suf}\ \phi_2$. The case for $\phi = \mathbf{f}\,\psi$ is similar to the base step and we omit here.

If $\phi = \phi_1\ \mathbf{suf}\ \phi_2$, let $Y = y \diamond Z$. Then from (A2), by applying $(\mathbf{suf})$ twice (firstly on the right and then on the left) and applying $(\vee R)$ and $(\vee L)$, we need to prove

(A2.1) $Z \diamond x : \phi_2 \Rightarrow Z \diamond x \diamond x : \phi_2, Z \diamond x \diamond x : \phi_1 \wedge \phi_1\ \mathbf{suf}\ \phi_2$, from which, by (wk) on the right, we obtain

$$(A2.11)\quad Z \diamond x : \phi_2 \Rightarrow Z \diamond x \diamond x : \phi_2.$$

By firstly replacing Z with Y (using (sub)) and then the induction hypothesis on ϕ_2, (A2.11) is provable.

(A2.2) $Z \diamond x : \phi_1 \wedge \phi_1\ \mathbf{suf}\ \phi_2 \Rightarrow Z \diamond x \diamond x : \phi_2, Z \diamond x \diamond x : \phi_1 \wedge \phi_1\ \mathbf{suf}\ \phi_2$.

By $(\wedge R)$ on $Z \diamond x \diamond x : \phi_1 \wedge \phi_1\ \mathbf{suf}\ \phi_2$, (A2.2) can be split into

(A2.21) $Z \diamond x : \phi_1 \wedge \phi_1\ \mathbf{suf}\ \phi_2 \Rightarrow Z \diamond x \diamond x : \phi_2, Z \diamond x \diamond x : \phi_1$, which can be easily proved by induction hypothesis on ϕ_1 after (wk) and $(\wedge L)$.

(A2.22) $Z \diamond x : \phi_1 \wedge \phi_1\ \mathbf{suf}\ \phi_2 \Rightarrow Z \diamond x \diamond x : \phi_1, Z \diamond x \diamond x : \phi_1\ \mathbf{suf}\ \phi_2$, from which, after $(\wedge L)$ and (wk) on both sides, we obtain

$$(A2.221)\quad Z \diamond x : \phi_1\ \mathbf{suf}\ \phi_2 \Rightarrow Z \diamond x \diamond x : \phi_1\ \mathbf{suf}\ \phi_2.$$

(A2.221) is exactly (A2) after variable replacement. Moreover, from (A2) to (A2.221) the derivation path is progressive since rule $(\mathbf{suf})$ has been used.

□

Below we show that each special rule for process logic in PPL (Table 1) is provable in G3PPL. We firstly describe each of them as a proposition, then prove them (Proposition A.1 — A.15).

Proposition A.1 (Rule (i)). *Rule (i):* $\mathbf{f}(\phi \vee \psi) \leftrightarrow \mathbf{f}\,\phi \vee \mathbf{f}\,\psi$ *is derived in G3PPL.*

Proof. Let $X \in TVar$. We only prove

$$(i.a)\quad X : \mathbf{f}(\phi \vee \psi) \Rightarrow X : \mathbf{f}\,\phi \vee \mathbf{f}\,\psi.$$

The other direction is similar.

Without loss of generality, let $X = x \diamond Y$. Then by $(\mathbf{f})$, we need

$$(i.a1)\quad x : \phi \vee \psi \Rightarrow x : \phi, x : \psi,$$

which is proved trivially by $(\vee L)$ and (ax). □

Proposition A.2 (Rule (ii)). *Rule (ii):* $\mathbf{f}\,\neg\phi \leftrightarrow \neg\,\mathbf{f}\,\phi$ *is derived in G3PPL.*

Proof. Let $X \in TVar$. We only prove

$$(ii.a)\quad X : \mathbf{f}\,\neg\phi \Rightarrow X : \neg\,\mathbf{f}\,\phi.$$

The other direction is similar.

Without loss of generality, let $X = x \diamond Y$. By $(\mathbf{f})$ and $(\neg R)$ we obtain

$$x : \neg\phi, x : \phi \Rightarrow \cdot,$$

which is trivially proved by (ax). □

Proposition A.3 (Rule (iii)). *Rule (iii):* $(\phi \,\mathbf{suf}\, \psi) \vee (\phi \,\mathbf{suf}\, \chi) \leftrightarrow \phi \,\mathbf{suf}\, (\psi \vee \chi)$ *is derived in G3PPL.*

Proof. Let $X \in \mathit{TVar}$. We need to prove

(iii.a) $X : (\phi \,\mathbf{suf}\, \psi) \vee (\phi \,\mathbf{suf}\, \chi) \Rightarrow X : \phi \,\mathbf{suf}\, (\psi \vee \chi)$.
(iii.b) $X : \phi \,\mathbf{suf}\, (\psi \vee \chi) \Rightarrow X : (\phi \,\mathbf{suf}\, \psi) \vee (\phi \,\mathbf{suf}\, \chi)$.

(iii.a) is proved in Section 6, here we only prove (iii.b). That is to prove

$$(iii.b1)\;\; X : \phi \,\mathbf{suf}\, (\psi \vee \chi) \Rightarrow X : (\phi \,\mathbf{suf}\, \psi), X : (\phi \,\mathbf{suf}\, \chi)$$

by $(\vee R)$. Without loss of generality, let $X = x \diamond Y$. By applying $(\mathbf{suf})$ and then $(\vee L)$ on the right , we split (iii.b1) into

$(iii.b11)\;\; Y : \psi \vee \chi \Rightarrow x \diamond Y : (\phi \,\mathbf{suf}\, \psi), x \diamond Y : (\phi \,\mathbf{suf}\, \chi)$
$(iii.b12)\;\; Y : \phi \wedge \phi \,\mathbf{suf}\, (\psi \vee \chi) \Rightarrow x \diamond Y : (\phi \,\mathbf{suf}\, \psi), x \diamond Y : (\phi \,\mathbf{suf}\, \chi)$

For (iii.b11), by applying $(\mathbf{suf})$ on the right, we obtain

$$(iii.b111)\;\; Y : \psi \vee \chi \Rightarrow Y : \psi, Y : \phi \wedge \phi \,\mathbf{suf}\, \psi, Y : \chi, Y : \phi \wedge \phi \,\mathbf{suf}\, \chi.$$

Observing that since the right side contains both $Y : \psi$ and $Y : \chi$, (iii.b111) can be proved by (ax) for different cases on the right side.

For (iii.b12), by applying $(\mathbf{suf})$ on the right, we obtain

$$(iii.b121)\;\; Y : \phi \wedge \phi \,\mathbf{suf}\, (\psi \vee \chi) \Rightarrow Y : \psi, Y : \phi \wedge \phi \,\mathbf{suf}\, \psi, Y : \chi, Y : \phi \wedge \phi \,\mathbf{suf}\, \chi.$$

By $(\wedge R)$ on $Y : \phi \wedge \phi \,\mathbf{suf}\, \psi$ and $Y : \phi \wedge \phi \,\mathbf{suf}\, \chi$, we split the right side into several branches. Since both sides contain $Y : \phi$, all branches but the following one can be proved by directly applying (ax):

$$(iii.b1211)\;\; Y : \phi \wedge \phi \,\mathbf{suf}\, (\psi \vee \chi) \Rightarrow Y : \psi, Y : \phi \,\mathbf{suf}\, \psi, Y : \chi, Y : \phi \,\mathbf{suf}\, \chi.$$

From (iii.b1211), by (wk), we have

$$(iii.b12111)\;\; Y : \phi \,\mathbf{suf}\, (\psi \vee \chi) \Rightarrow Y : \phi \,\mathbf{suf}\, \psi, Y : \phi \,\mathbf{suf}\, \chi,$$

which is exactly (iii.b1) (replacing Y with X). The derivation from (iii.b1) to (iii.b12111) is progressive since rule $(\mathbf{suf})$ is applied. This makes the whole preproof a cyclic proof.

□

Proposition A.4 (Rule (iv)). *Rule (iv):* $(\phi \,\mathbf{suf}\, \psi) \wedge (\chi \,\mathbf{suf}\, \omega) \leftrightarrow (\phi \wedge \chi) \,\mathbf{suf}\, (\psi \wedge \omega) \vee (\phi \wedge \chi) \,\mathbf{suf}\, (\phi \wedge \omega \wedge \phi \,\mathbf{suf}\, \psi) \vee (\phi \wedge \chi) \,\mathbf{suf}\, (\psi \wedge \chi \wedge \chi \,\mathbf{suf}\, \omega)$ *is derived in G3PPL.*

Proof. Let $X \in \mathit{TVar}$, we prove that

$(iv.a)\;\; X : (\phi \,\mathbf{suf}\, \psi) \wedge (\chi \,\mathbf{suf}\, \omega) \Rightarrow X : (\phi \wedge \chi) \,\mathbf{suf}\, (\psi \wedge \omega) \vee (\phi \wedge \chi) \,\mathbf{suf}\, (\phi \wedge \omega \wedge \phi \,\mathbf{suf}\, \psi) \vee (\phi \wedge \chi) \,\mathbf{suf}\, (\psi \wedge \chi \wedge \chi \,\mathbf{suf}\, \omega)$.
$(iv.b)\;\; X : (\phi \wedge \chi) \,\mathbf{suf}\, (\psi \wedge \omega) \vee (\phi \wedge \chi) \,\mathbf{suf}\, (\phi \wedge \omega \wedge \phi \,\mathbf{suf}\, \psi) \vee (\phi \wedge \chi) \,\mathbf{suf}\, (\psi \wedge \chi \wedge \chi \,\mathbf{suf}\, \omega) \Rightarrow X : (\phi \,\mathbf{suf}\, \psi) \wedge (\chi \,\mathbf{suf}\, \omega)$.

Case for proving $(iv.a)$: Let

$$A =_{df} X : (\phi \wedge \chi) \,\mathbf{suf}\, (\psi \wedge \omega),$$
$$B =_{df} X : (\phi \wedge \chi) \,\mathbf{suf}\, (\phi \wedge \omega \wedge \phi \,\mathbf{suf}\, \psi),$$
$$C =_{df} X : (\phi \wedge \chi) \,\mathbf{suf}\, (\psi \wedge \chi \wedge \chi \,\mathbf{suf}\, \omega).$$

By applying the rules $(\wedge L)$ and $(\vee R)$ on both sides respectively, we obtain the sequent:

$$(iv.a1)\ \ X : \phi \,\mathbf{suf}\, \psi, X : \chi \,\mathbf{suf}\, \omega \Rightarrow A, B, C,$$

Let $X = y \diamond Y$, by applying the rule $(\mathbf{suf})$ on both sides, we have

$$(iv.a11)\ Y : \psi \vee (\phi \wedge \phi \,\mathbf{suf}\, \psi), Y : \omega \vee (\chi \wedge \chi \,\mathbf{suf}\, \omega) \Rightarrow A_1, A_2, B_1, B_2, C_1, C_2,$$

where let

$$\begin{aligned}
A_1 &=_{df} Y : \psi \wedge \omega, && \text{obtained from } A\\
A_2 &=_{df} Y : \phi \wedge \chi \wedge (\phi \wedge \chi) \,\mathbf{suf}\, (\psi \wedge \omega), && \text{obtained from } A\\
B_1 &=_{df} Y : \phi \wedge \omega \wedge \phi \,\mathbf{suf}\, \psi, && \text{obtained from } B\\
B_2 &=_{df} Y : (\phi \wedge \chi) \wedge (\phi \wedge \chi) \,\mathbf{suf}\, (\phi \wedge \omega \wedge \phi \,\mathbf{suf}\, \psi), && \text{obtained from } B\\
C_1 &=_{df} Y : \psi \wedge \chi \wedge \chi \,\mathbf{suf}\, \omega, && \text{obtained from } C\\
C_2 &=_{df} Y : (\phi \wedge \chi) \wedge (\phi \wedge \chi) \,\mathbf{suf}\, (\psi \wedge \chi \wedge \chi \,\mathbf{suf}\, \omega), && \text{obtained from } C
\end{aligned}$$

By the rule $(\vee L)$, we split (iv.a11) into 4 cases:

(iv.a111) $Y : \psi, Y : \omega \Rightarrow A_1, A_2, B_1, B_2, C_1, C_2$
(iv.a112) $Y : (\phi \wedge \phi \,\mathbf{suf}\, \psi), Y : \omega \Rightarrow A_1, A_2, B_1, B_2, C_1, C_2$
(iv.a113) $Y : \psi, Y : (\chi \wedge \chi \,\mathbf{suf}\, \omega) \Rightarrow A_1, A_2, B_1, B_2, C_1, C_2$
(iv.a114) $Y : (\phi \wedge \phi \,\mathbf{suf}\, \psi), Y : (\chi \wedge \chi \,\mathbf{suf}\, \omega) \Rightarrow A_1, A_2, B_1, B_2, C_1, C_2$

The proofs of cases (iv.a111), (iv.a112), (iv.a113) are trivial and are all terminal branches (without buds), since in each of these cases a formula on the left side directly matches one of the right-side disjuncts A_1, A_2, B_1, B_2, C_1, or C_2 up to logical connectives, allowing the branch to be closed by $(\vee R)$, $(\wedge L)$, and (ax) without any application of rule $(\mathbf{suf})$.

For case (iv.a114), by applying the rule $(\wedge R)$ on A_2, B_2, C_2 respectively, we obtain two parts of proof branches, named (iv.a1141) and (iv.a1142) respectively. Part (iv.a1141) contains 3 sequents of the form:

$$Y : (\phi \wedge \phi \,\mathbf{suf}\, \psi), Y : (\chi \wedge \chi \,\mathbf{suf}\, \omega) \Rightarrow ..., Y : (\phi \wedge \chi), ...,$$

which can be easily proved by applying the rule $(\wedge L)$ on the left and the rule (ax). Part (iv.a1142) is the sequent of the form:

$$Y : (\phi \wedge \phi \,\mathbf{suf}\, \psi), Y : (\chi \wedge \chi \,\mathbf{suf}\, \omega) \Rightarrow ..., Y : A, Y : B, Y : C, ...$$

By applying the rule $(\wedge L)$, the rule (wk) on both sides for several times, we can get

$$(iv.a11421)\ \ Y : \phi \,\mathbf{suf}\, \psi, Y : \chi \,\mathbf{suf}\, \omega \Rightarrow Y : A, Y : B, Y : C,$$

which is exactly (iv.a1). Note that along the derivation path from (iv.a1) to (iv.a11421), rule $(\mathbf{suf})$ has been applied, which guarantees the whole preproof is a cyclic proof.

Case for proving $(iv.b)$: By applying the rules $(\vee L)$ and $(\wedge R)$ on the left and right sides of the sequent respectively, we split (iv.b) into 6 cases as follows:

(iv.b1) $X : (\phi \wedge \chi) \,\mathbf{suf}\, (\psi \wedge \omega) \Rightarrow X : (\phi \,\mathbf{suf}\, \psi)$
(iv.b2) $X : (\phi \wedge \chi) \,\mathbf{suf}\, (\phi \wedge \omega \wedge \phi \,\mathbf{suf}\, \psi) \Rightarrow X : (\phi \,\mathbf{suf}\, \psi)$
(iv.b3) $X : (\phi \wedge \chi) \,\mathbf{suf}\, (\psi \wedge \chi \wedge \chi \,\mathbf{suf}\, \omega) \Rightarrow X : (\phi \,\mathbf{suf}\, \psi)$
(iv.b4) $X : (\phi \wedge \chi) \,\mathbf{suf}\, (\psi \wedge \omega) \Rightarrow X : (\chi \,\mathbf{suf}\, \omega)$

(iv.b5) $X : (\phi \wedge \chi)\ \mathbf{suf}(\phi \wedge \omega \wedge \phi\ \mathbf{suf}\ \psi) \Rightarrow X : (\chi\ \mathbf{suf}\ \omega)$
(iv.b6) $X : (\phi \wedge \chi)\ \mathbf{suf}(\psi \wedge \chi \wedge \chi\ \mathbf{suf}\ \omega) \Rightarrow X : (\chi\ \mathbf{suf}\ \omega)$

Case (iv.b1): Without loss of generality, let $X = y \diamond Y$. From (iv.b1), by applying the rule ($\mathbf{suf}$), we can have

$$(iv.b11)\ \ Y : (\psi \wedge \omega) \vee (\phi \wedge \chi) \wedge (\phi \wedge \chi)\ \mathbf{suf}(\psi \wedge \omega) \Rightarrow y \diamond Y : (\phi\ \mathbf{suf}\ \psi)$$

From (iv.b11), we apply the rule ($\vee L$), and obtain:

$$(iv.b111)\ \ Y : (\psi \wedge \omega) \Rightarrow y \diamond Y : (\phi\ \mathbf{suf}\ \psi)$$
$$(iv.b112)\ \ Y : (\phi \wedge \chi) \wedge (\phi \wedge \chi)\ \mathbf{suf}(\psi \wedge \omega) \Rightarrow y \diamond Y : (\phi\ \mathbf{suf}\ \psi)$$

The case for (iv.b111) is trivial (just extend $y \diamond Y : (\phi\ \mathbf{suf}\ \psi)$ on the right side by the rule ($\mathbf{suf}$) and can obtain the same item $Y : \psi$ as on the left side), we omit it. For case (iv.b112), we extending $y \diamond Y : (\phi\ \mathbf{suf}\ \psi)$ with the rule ($\mathbf{suf}$) and other rules on the left side, we obtain:

$$(iv.b1121)\ \ Y : (\phi \wedge \chi), Y : (\phi \wedge \chi)\ \mathbf{suf}(\psi \wedge \omega) \Rightarrow Y : \phi,$$
$$(iv.b1122)\ \ Y : (\phi \wedge \chi), Y : (\phi \wedge \chi)\ \mathbf{suf}(\psi \wedge \omega) \Rightarrow Y : \phi\ \mathbf{suf}\ \psi$$

where (iv.b1121) is obvious (omitted). From (iv.b1122), we obtain by applying the rule (wk):

$$(iv.b11221)\ \ Y : (\phi \wedge \chi)\ \mathbf{suf}(\psi \wedge \omega) \Rightarrow Y : \phi\ \mathbf{suf}\ \psi$$

which is exactly (iv.b1), forming a back-link. The derivation path from (iv.b1) to (iv.b11221) is progressive since rule ($\mathbf{suf}$) is applied.

Case (iv.b2): Without loss of generality, let $X = y \diamond Y$. From (iv.b2), by applying the rule ($\mathbf{suf}$), we can have

$$(iv.b21)\ \ Y : (\phi \wedge \omega \wedge \phi\ \mathbf{suf}\ \psi) \vee (\phi \wedge \chi) \wedge (\phi \wedge \chi)\ \mathbf{suf}(\phi \wedge \omega \wedge \phi\ \mathbf{suf}\ \psi) \Rightarrow y \diamond Y : (\phi\ \mathbf{suf}\ \psi)$$

From (iv.b21), we apply the rule ($\vee L$), and obtain:

$$(iv.b211)\ \ Y : (\phi \wedge \omega \wedge \phi\ \mathbf{suf}\ \psi) \Rightarrow y \diamond Y : (\phi\ \mathbf{suf}\ \psi)$$
$$(iv.b212)\ \ Y : (\phi \wedge \chi) \wedge (\phi \wedge \chi)\ \mathbf{suf}(\phi \wedge \omega \wedge \phi\ \mathbf{suf}\ \psi) \Rightarrow y \diamond Y : (\phi\ \mathbf{suf}\ \psi)$$

For case (iv.b211), by extending $y \diamond Y : (\phi\ \mathbf{suf}\ \psi)$ on the right side by the rule ($\mathbf{suf}$) and ($\wedge R$), we need to show $Y : \psi$, $Y : \phi$ and $Y : \phi\ \mathbf{suf}\ \psi$; all obtainable from the left side by ($\wedge L$) and (ax). We omit it. For case (iv.b212), by extending $y \diamond Y : (\phi\ \mathbf{suf}\ \psi)$ with the rule ($\mathbf{suf}$) and other rules on the left side, we obtain:

$$(iv.b2121)\ \ Y : (\phi \wedge \chi), Y : (\phi \wedge \chi)\ \mathbf{suf}(\phi \wedge \omega \wedge \phi\ \mathbf{suf}\ \psi) \Rightarrow Y : \phi,$$
$$(iv.b2122)\ \ Y : (\phi \wedge \chi), Y : (\phi \wedge \chi)\ \mathbf{suf}(\phi \wedge \omega \wedge \phi\ \mathbf{suf}\ \psi) \Rightarrow Y : \phi\ \mathbf{suf}\ \psi$$

where (iv.b2121) is obvious (omitted). From (iv.b2122), we obtain by applying the rule (wk):

$$(iv.b21221)\ \ Y : (\phi \wedge \chi)\ \mathbf{suf}(\phi \wedge \omega \wedge \phi\ \mathbf{suf}\ \psi) \Rightarrow Y : \phi\ \mathbf{suf}\ \psi$$

which is exactly (iv.b2), forming a back-link. The derivation path from (iv.b2) to (iv.b21221) is progressive since rule ($\mathbf{suf}$) is applied.

Case (iv.b3): Without loss of generality, let $X = y \diamond Y$. From (iv.b3), by applying the rule ($\mathbf{suf}$), we can have

$$(iv.b31)\ \ Y : (\psi \wedge \chi \wedge \chi\ \mathbf{suf}\ \omega) \vee (\phi \wedge \chi) \wedge (\phi \wedge \chi)\ \mathbf{suf}(\psi \wedge \chi \wedge \chi\ \mathbf{suf}\ \omega) \Rightarrow y \diamond Y : (\phi\ \mathbf{suf}\ \psi)$$

From (iv.b31), we apply the rule $(\vee L)$, and obtain:

$$(iv.b311)\ \ Y : (\psi \wedge \chi \wedge \chi\ \mathbf{suf}\ \omega) \Rightarrow y \diamond Y : (\phi\ \mathbf{suf}\ \psi)$$

$$(iv.b312)\ \ Y : (\phi \wedge \chi) \wedge (\phi \wedge \chi)\ \mathbf{suf} (\psi \wedge \chi \wedge \chi\ \mathbf{suf}\ \omega) \Rightarrow y \diamond Y : (\phi\ \mathbf{suf}\ \psi)$$

The case for (iv.b311) is trivial (just extend $y \diamond Y : (\phi\ \mathbf{suf}\ \psi)$ on the right side by the rule $(\mathbf{suf})$ and can obtain the same item $Y : \psi$ as on the left side), we omit it. For case (iv.b312), by extending $y \diamond Y : (\phi\ \mathbf{suf}\ \psi)$ with the rule $(\mathbf{suf})$ and other rules on the left side, we obtain:

$$(iv.b3121)\ \ Y : (\phi \wedge \chi), Y : (\phi \wedge \chi)\ \mathbf{suf} (\psi \wedge \chi \wedge \chi\ \mathbf{suf}\ \omega) \Rightarrow Y : \phi,$$

$$(iv.b3122)\ \ Y : (\phi \wedge \chi), Y : (\phi \wedge \chi)\ \mathbf{suf} (\psi \wedge \chi \wedge \chi\ \mathbf{suf}\ \omega) \Rightarrow Y : \phi\ \mathbf{suf}\ \psi$$

where (iv.b3121) is obvious (omitted). From (iv.b3122), we obtain by applying the rule (wk):

$$(iv.b31221)\ \ Y : (\phi \wedge \chi)\ \mathbf{suf} (\psi \wedge \chi \wedge \chi\ \mathbf{suf}\ \omega) \Rightarrow Y : \phi\ \mathbf{suf}\ \psi$$

which is exactly (iv.b3), forming a back-link. The derivation path from (iv.b3) to (iv.b31221) is progressive since rule $(\mathbf{suf})$ is applied.

Case (iv.b4): Without loss of generality, let $X = y \diamond Y$. From (iv.b4), by applying the rule $(\mathbf{suf})$, we can have

$$(iv.b41)\ \ Y : (\psi \wedge \omega) \vee (\phi \wedge \chi) \wedge (\phi \wedge \chi)\ \mathbf{suf} (\psi \wedge \omega) \Rightarrow y \diamond Y : (\chi\ \mathbf{suf}\ \omega)$$

From (iv.b41), we apply the rule $(\vee L)$, and obtain:

$$(iv.b411)\ \ Y : (\psi \wedge \omega) \Rightarrow y \diamond Y : (\chi\ \mathbf{suf}\ \omega)$$

$$(iv.b412)\ \ Y : (\phi \wedge \chi) \wedge (\phi \wedge \chi)\ \mathbf{suf} (\psi \wedge \omega) \Rightarrow y \diamond Y : (\chi\ \mathbf{suf}\ \omega)$$

The case for (iv.b411) is trivial (just extend $y \diamond Y : (\chi\ \mathbf{suf}\ \omega)$ on the right side by the rule $(\mathbf{suf})$ and can obtain the same item $Y : \omega$ as on the left side), we omit it. For case (iv.b412), by extending $y \diamond Y : (\chi\ \mathbf{suf}\ \omega)$ with the rule $(\mathbf{suf})$ and other rules on the left side, we obtain:

$$(iv.b4121)\ \ Y : (\phi \wedge \chi), Y : (\phi \wedge \chi)\ \mathbf{suf} (\psi \wedge \omega) \Rightarrow Y : \chi,$$

$$(iv.b4122)\ \ Y : (\phi \wedge \chi), Y : (\phi \wedge \chi)\ \mathbf{suf} (\psi \wedge \omega) \Rightarrow Y : \chi\ \mathbf{suf}\ \omega$$

where (iv.b4121) is obvious (omitted). From (iv.b4122), we obtain by applying the rule (wk):

$$(iv.b41221)\ \ Y : (\phi \wedge \chi)\ \mathbf{suf} (\psi \wedge \omega) \Rightarrow Y : \chi\ \mathbf{suf}\ \omega$$

which is exactly (iv.b4), forming a back-link. The derivation path from (iv.b4) to (iv.b41221) is progressive since rule $(\mathbf{suf})$ is applied.

Case (iv.b5): Without loss of generality, let $X = y \diamond Y$. From (iv.b5), by applying the rule $(\mathbf{suf})$, we can have

$$(iv.b51)\ \ Y : (\phi \wedge \omega \wedge \phi\ \mathbf{suf}\ \psi) \vee (\phi \wedge \chi) \wedge (\phi \wedge \chi)\ \mathbf{suf} (\phi \wedge \omega \wedge \phi\ \mathbf{suf}\ \psi) \Rightarrow y \diamond Y : (\chi\ \mathbf{suf}\ \omega)$$

From (iv.b51), we apply the rule $(\vee L)$, and obtain:

$$(iv.b511)\ \ Y : (\phi \wedge \omega \wedge \phi\ \mathbf{suf}\ \psi) \Rightarrow y \diamond Y : (\chi\ \mathbf{suf}\ \omega)$$

$$(iv.b512)\ \ Y : (\phi \wedge \chi) \wedge (\phi \wedge \chi)\ \mathbf{suf} (\phi \wedge \omega \wedge \phi\ \mathbf{suf}\ \psi) \Rightarrow y \diamond Y : (\chi\ \mathbf{suf}\ \omega)$$

The case for (iv.b511) is trivial (just extend $y \diamond Y : (\chi\ \mathbf{suf}\ \omega)$ on the right side by the rule $(\mathbf{suf})$ and can obtain the same item $Y : \omega$ as on the left side), we omit it. For case (iv.b512), by extending $y \diamond Y : (\chi\ \mathbf{suf}\ \omega)$ with the rule $(\mathbf{suf})$ and other

rules on the left side, we obtain:

$$(iv.b5121)\ \ Y : (\phi \wedge \chi), Y : (\phi \wedge \chi)\ \mathbf{suf}(\phi \wedge \omega \wedge \phi\ \mathbf{suf}\ \psi) \Rightarrow Y : \chi,$$

$$(iv.b5122)\ \ Y : (\phi \wedge \chi), Y : (\phi \wedge \chi)\ \mathbf{suf}(\phi \wedge \omega \wedge \phi\ \mathbf{suf}\ \psi) \Rightarrow Y : \chi\ \mathbf{suf}\ \omega$$

where (iv.b5121) is obvious (omitted). From (iv.b5122), we obtain by applying the rule (wk):

$$(iv.b51221)\ \ Y : (\phi \wedge \chi)\ \mathbf{suf}(\phi \wedge \omega \wedge \phi\ \mathbf{suf}\ \psi) \Rightarrow Y : \chi\ \mathbf{suf}\ \omega$$

which is exactly (iv.b5), forming a back-link. The derivation path from (iv.b5) to (iv.b51221) is progressive since rule (**suf**) is applied.

Case (iv.b6): Without loss of generality, let $X = y \diamond Y$. From (iv.b6), by applying the rule (**suf**), we can have

$$(iv.b61)\ \ Y : (\psi \wedge \chi \wedge \chi\ \mathbf{suf}\ \omega) \vee (\phi \wedge \chi) \wedge (\phi \wedge \chi)\ \mathbf{suf}(\psi \wedge \chi \wedge \chi\ \mathbf{suf}\ \omega) \Rightarrow y \diamond Y : (\chi\ \mathbf{suf}\ \omega)$$

From (iv.b61), we apply the rule $(\vee L)$, and obtain:

$$(iv.b611)\ \ Y : (\psi \wedge \chi \wedge \chi\ \mathbf{suf}\ \omega) \Rightarrow y \diamond Y : (\chi\ \mathbf{suf}\ \omega)$$

$$(iv.b612)\ \ Y : (\phi \wedge \chi) \wedge (\phi \wedge \chi)\ \mathbf{suf}(\psi \wedge \chi \wedge \chi\ \mathbf{suf}\ \omega) \Rightarrow y \diamond Y : (\chi\ \mathbf{suf}\ \omega)$$

For case (iv.b611), by extending $y \diamond Y : (\chi\ \mathbf{suf}\ \omega)$ on the right side by the rule (**suf**) and $(\wedge R)$, we need to show $Y : \omega$, $Y : \chi$ and $Y : \chi\ \mathbf{suf}\ \omega$; all obtainable from the left side by $(\wedge L)$ and (ax). We omit it. For case (iv.b612), by extending $y \diamond Y : (\chi\ \mathbf{suf}\ \omega)$ with the rule (**suf**) and other rules on the left side, we obtain:

$$(iv.b6121)\ \ Y : (\phi \wedge \chi), Y : (\phi \wedge \chi)\ \mathbf{suf}(\psi \wedge \chi \wedge \chi\ \mathbf{suf}\ \omega) \Rightarrow Y : \chi,$$

$$(iv.b6122)\ \ Y : (\phi \wedge \chi), Y : (\phi \wedge \chi)\ \mathbf{suf}(\psi \wedge \chi \wedge \chi\ \mathbf{suf}\ \omega) \Rightarrow Y : \chi\ \mathbf{suf}\ \omega$$

where (iv.b6121) is obvious (omitted). From (iv.b6122), we obtain by applying the rule (wk):

$$(iv.b61221)\ \ Y : (\phi \wedge \chi)\ \mathbf{suf}(\psi \wedge \chi \wedge \chi\ \mathbf{suf}\ \omega) \Rightarrow Y : \chi\ \mathbf{suf}\ \omega$$

which is exactly (iv.b6), forming a back-link. The derivation path from (iv.b6) to (iv.b61221) is progressive since rule (**suf**) is applied.

From these 6 cases, it is not hard to see that the whole derivation from (iv.b) is a cyclic proof. Thus we conclude the proof of (iv.b).

□

Proposition A.5 (Rule (v)). *Rule (v):* $\neg(\phi\ \mathbf{suf}\ \psi) \leftrightarrow \neg(\mathit{true}\ \mathbf{suf}\ \psi) \vee (\neg\psi)\ \mathbf{suf}(\neg\phi \wedge \neg\psi)$ *is derived in G3PPL.*

Proof. Let $X \in \mathit{TVar}$. We need to prove

$(v.a)\ \ X : \neg(\phi\ \mathbf{suf}\ \psi) \Rightarrow X : \neg(\mathit{true}\ \mathbf{suf}\ \psi) \vee (\neg\psi)\ \mathbf{suf}(\neg\phi \wedge \neg\psi)$.

$(v.b)\ \ X : \neg(\mathit{true}\ \mathbf{suf}\ \psi) \vee (\neg\psi)\ \mathbf{suf}(\neg\phi \wedge \neg\psi) \Rightarrow X : \neg(\phi\ \mathbf{suf}\ \psi)$.

Case for proving $(v.a)$: By applying $(\neg L)$ on the left and $(\vee R)$ on the right, we obtain the sequent:

$$(v.a1)\ \ X : (\mathit{true}\ \mathbf{suf}\ \psi) \Rightarrow X : (\phi\ \mathbf{suf}\ \psi), X : (\neg\psi)\ \mathbf{suf}(\neg\phi \wedge \neg\psi)$$

Without loss of generality, let $X = y \diamond Y$. By applying the rule (**suf**) on $X : (\mathit{true}\ \mathbf{suf}\ \psi)$, we have:

$$(v.a11)\ \ Y : \psi \vee (\mathit{true} \wedge \mathit{true}\ \mathbf{suf}\ \psi) \Rightarrow y \diamond Y : (\phi\ \mathbf{suf}\ \psi), y \diamond Y : (\neg\psi)\ \mathbf{suf}(\neg\phi \wedge \neg\psi)$$

By the rule $(\vee L)$, we split (v.a11) into 2 cases:

(v.a111) $Y : \psi \Rightarrow y \diamond Y : (\phi \,\mathbf{suf}\, \psi), y \diamond Y : (\neg\psi)\,\mathbf{suf}(\neg\phi \wedge \neg\psi)$

(v.a112) $Y : \mathit{true} \wedge \mathit{true}\,\mathbf{suf}\,\psi \Rightarrow y \diamond Y : (\phi \,\mathbf{suf}\, \psi), y \diamond Y : (\neg\psi)\,\mathbf{suf}(\neg\phi \wedge \neg\psi)$

Case (v.a111) is trivial (just extend $y \diamond Y : (\phi \,\mathbf{suf}\, \psi)$ on the right side by the rule (**suf**) and can obtain the same item $Y : \psi$ as on the left side), we omit it.

For case (v.a112), by $(\wedge L)$ and weakening, it reduces to:

$$Y : \mathit{true}\,\mathbf{suf}\,\psi \Rightarrow y \diamond Y : (\phi \,\mathbf{suf}\, \psi), y \diamond Y : (\neg\psi)\,\mathbf{suf}(\neg\phi \wedge \neg\psi)$$

By applying (**suf**) on $y \diamond Y : (\phi \,\mathbf{suf}\, \psi)$ and $y \diamond Y : (\neg\psi)\,\mathbf{suf}(\neg\phi \wedge \neg\psi)$ on the right side, we obtain the succedent items:

$D_1 =_{df} Y : \psi$, obtained from $y \diamond Y : (\phi \,\mathbf{suf}\, \psi)$

$D_2 =_{df} Y : \phi \wedge (\phi \,\mathbf{suf}\, \psi)$, obtained from $y \diamond Y : (\phi \,\mathbf{suf}\, \psi)$

$D_3 =_{df} Y : (\neg\phi \wedge \neg\psi)$, obtained from $y \diamond Y : (\neg\psi)\,\mathbf{suf}(\neg\phi \wedge \neg\psi)$

$D_4 =_{df} Y : \neg\psi \wedge (\neg\psi)\,\mathbf{suf}(\neg\phi \wedge \neg\psi)$, obtained from $y \diamond Y : (\neg\psi)\,\mathbf{suf}(\neg\phi \wedge \neg\psi)$

So we need to show:

$$(v.a1121)\quad Y : \mathit{true}\,\mathbf{suf}\,\psi \Rightarrow D_1, D_2, D_3, D_4$$

By $(\wedge R)$ on D_2 and D_4, the non-trivial remaining obligation after closing the branches for $Y : \phi$, $Y : \neg\psi$, etc. by the rules $(\wedge R)$, $(\neg R)$, (ax) is:

$$Y : \mathit{true}\,\mathbf{suf}\,\psi \Rightarrow Y : \psi, Y : \phi \,\mathbf{suf}\, \psi, Y : (\neg\phi \wedge \neg\psi), Y : (\neg\psi)\,\mathbf{suf}(\neg\phi \wedge \neg\psi)$$

By weakening, this reduces to:

$$(v.a11211)\quad Y : \mathit{true}\,\mathbf{suf}\,\psi \Rightarrow Y : \phi \,\mathbf{suf}\, \psi, Y : (\neg\psi)\,\mathbf{suf}(\neg\phi \wedge \neg\psi)$$

which is exactly (v.a1), forming a back-link. The derivation path from (v.a1) to (v.a11211) is progressive since rule (**suf**) is applied.

Case for proving $(v.b)$: By applying $(\vee L)$ on the left, we split it into 2 cases:

(v.b1) $X : \neg(\mathit{true}\,\mathbf{suf}\,\psi) \Rightarrow X : \neg(\phi \,\mathbf{suf}\, \psi)$

(v.b2) $X : (\neg\psi)\,\mathbf{suf}(\neg\phi \wedge \neg\psi) \Rightarrow X : \neg(\phi \,\mathbf{suf}\, \psi)$

Case (v.b1): By $(\neg L)$ and $(\neg R)$:

$$(v.b11)\quad X : \phi \,\mathbf{suf}\, \psi \Rightarrow X : \mathit{true}\,\mathbf{suf}\,\psi$$

Without loss of generality, let $X = y \diamond Y$. By applying (**suf**) on the left, we have:

$$Y : \psi \vee (\phi \wedge \phi \,\mathbf{suf}\, \psi) \Rightarrow y \diamond Y : \mathit{true}\,\mathbf{suf}\,\psi$$

By $(\vee L)$:

(v.b111) $Y : \psi \Rightarrow y \diamond Y : \mathit{true}\,\mathbf{suf}\,\psi$

(v.b112) $Y : \phi \wedge \phi \,\mathbf{suf}\, \psi \Rightarrow y \diamond Y : \mathit{true}\,\mathbf{suf}\,\psi$

Case (v.b111) is trivial (extend $y \diamond Y : \mathit{true}\,\mathbf{suf}\,\psi$ by (**suf**) to get $Y : \psi$), we omit it. For (v.b112), by $(\wedge L)$ and extending $y \diamond Y : \mathit{true}\,\mathbf{suf}\,\psi$ by (**suf**) and $(\wedge R)$, we have $Y : \psi$ and $Y : \mathit{true} \wedge \mathit{true}\,\mathbf{suf}\,\psi$ on the right side. The non-trivial branch after weakening yields:

$$(v.b1121)\quad Y : \phi \,\mathbf{suf}\, \psi \Rightarrow Y : \mathit{true}\,\mathbf{suf}\,\psi$$

which is exactly (v.b11), forming a back-link. The derivation path is progressive since rule (**suf**) is applied.

Case (v.b2): By $(\neg R)$:

$$(v.b21)\ \ X : (\neg\psi)\,\mathbf{suf}(\neg\phi \wedge \neg\psi), X : \phi\,\mathbf{suf}\,\psi \Rightarrow \cdot$$

Without loss of generality, let $X = y \diamond Y$. By applying (**suf**) on both formulas on the left, we have:

$$Y : (\neg\phi \wedge \neg\psi) \vee (\neg\psi \wedge (\neg\psi)\,\mathbf{suf}(\neg\phi \wedge \neg\psi)),\ \ Y : \psi \vee (\phi \wedge \phi\,\mathbf{suf}\,\psi) \Rightarrow \cdot$$

By $(\vee L)$, we split into 4 sub-cases:

(v.b2111) $Y : \neg\phi \wedge \neg\psi,\ Y : \psi \Rightarrow \cdot$

(v.b2112) $Y : \neg\phi \wedge \neg\psi,\ Y : \phi \wedge \phi\,\mathbf{suf}\,\psi \Rightarrow \cdot$

(v.b2113) $Y : \neg\psi \wedge (\neg\psi)\,\mathbf{suf}(\neg\phi \wedge \neg\psi),\ Y : \psi \Rightarrow \cdot$

(v.b2114) $Y : \neg\psi \wedge (\neg\psi)\,\mathbf{suf}(\neg\phi \wedge \neg\psi),\ Y : \phi \wedge \phi\,\mathbf{suf}\,\psi \Rightarrow \cdot$

Case (v.b2111): By $(\wedge L)$, we have $Y : \neg\psi, Y : \psi \Rightarrow \cdot$, which closes by $(\neg L)$ and (ax).

Case (v.b2112): By $(\wedge L)$, we have $Y : \neg\phi, Y : \phi \Rightarrow \cdot$, which closes by $(\neg L)$ and (ax).

Case (v.b2113): By $(\wedge L)$, we have $Y : \neg\psi, Y : \psi \Rightarrow \cdot$, which closes by $(\neg L)$ and (ax).

Case (v.b2114): By $(\wedge L)$ on both conjunctions:

$$Y : \neg\psi, Y : (\neg\psi)\,\mathbf{suf}(\neg\phi \wedge \neg\psi), Y : \phi, Y : \phi\,\mathbf{suf}\,\psi \Rightarrow \cdot$$

By weakening:

$$(v.b21141)\ \ Y : (\neg\psi)\,\mathbf{suf}(\neg\phi \wedge \neg\psi), Y : \phi\,\mathbf{suf}\,\psi \Rightarrow \cdot$$

which is exactly (v.b21), forming a back-link. The derivation path is progressive since rule (**suf**) is applied.

□

Proposition A.6 (Rule (vi)). *Rule (vi): $\phi\,\mathbf{suf}\,\psi \leftrightarrow \mathbf{n}\,\psi \vee \mathbf{n}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$ is derived in G3PPL.*

Proof. Let $X \in TVar$. Recall that $\mathbf{n}\,\theta =_{df} false\,\mathbf{suf}\,\theta$. We need to prove

$(vi.a)$ $X : \phi\,\mathbf{suf}\,\psi \Rightarrow X : \mathbf{n}\,\psi \vee \mathbf{n}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$.

$(vi.b)$ $X : \mathbf{n}\,\psi \vee \mathbf{n}(\phi \wedge (\phi\,\mathbf{suf}\,\psi)) \Rightarrow X : \phi\,\mathbf{suf}\,\psi$.

Case for proving $(vi.a)$: By applying $(\vee R)$, we need:

$$(vi.a1)\ \ X : \phi\,\mathbf{suf}\,\psi \Rightarrow X : false\,\mathbf{suf}\,\psi, X : false\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$$

Without loss of generality, let $X = y \diamond Y$. By applying (**suf**) on $X : \phi\,\mathbf{suf}\,\psi$ on the left, we have:

$$Y : \psi \vee (\phi \wedge \phi\,\mathbf{suf}\,\psi) \Rightarrow y \diamond Y : false\,\mathbf{suf}\,\psi, y \diamond Y : false\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$$

By (**suf**) on $y \diamond Y : false\,\mathbf{suf}\,\psi$ on the right, we get $Y : \psi$ and $Y : false \wedge false\,\mathbf{suf}\,\psi$; and by (**suf**) on $y \diamond Y : false\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$ on the right, we get $Y : \phi \wedge (\phi\,\mathbf{suf}\,\psi)$ and $Y : false \wedge false\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$. The items containing $false \wedge (\cdot)$ cannot be proved (since they contain *false*), so on the right side we can just leave $Y : \psi$ and $Y : \phi \wedge (\phi\,\mathbf{suf}\,\psi)$ by weakening.

By $(\vee L)$ on the left, we split into 2 cases:

(vi.a11) $Y : \psi \Rightarrow Y : \psi, Y : \phi \wedge (\phi\,\mathbf{suf}\,\psi)$

(vi.a12) $Y : \phi \wedge \phi\,\mathbf{suf}\,\psi \Rightarrow Y : \psi, Y : \phi \wedge (\phi\,\mathbf{suf}\,\psi)$

Case (vi.a11) is immediately closed by (ax). Case (vi.a12) is immediately closed by (ax).

Case for proving $(vi.b)$: By $(\vee L)$, we split into 2 cases:

(vi.b1) $X : \mathit{false}\,\mathbf{suf}\,\psi \Rightarrow X : \phi\,\mathbf{suf}\,\psi$
(vi.b2) $X : \mathit{false}\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi)) \Rightarrow X : \phi\,\mathbf{suf}\,\psi$

Case (vi.b1): Without loss of generality, let $X = y \diamond Y$. By $(\mathbf{suf})$ on the left:

$$Y : \psi \vee (\mathit{false} \wedge \mathit{false}\,\mathbf{suf}\,\psi) \Rightarrow y \diamond Y : \phi\,\mathbf{suf}\,\psi$$

Since $\mathit{false} \wedge (\cdot)$ is trivially false, by $(\vee L)$ this reduces to:

$$Y : \psi \Rightarrow y \diamond Y : \phi\,\mathbf{suf}\,\psi$$

By $(\mathbf{suf})$ on the right side, we obtain the disjunct $Y : \psi$, which is on the left side. This is trivially closed.

Case (vi.b2): Without loss of generality, let $X = y \diamond Y$. By $(\mathbf{suf})$ on the left:

$$Y : (\phi \wedge (\phi\,\mathbf{suf}\,\psi)) \vee (\mathit{false} \wedge \mathit{false}\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))) \Rightarrow y \diamond Y : \phi\,\mathbf{suf}\,\psi$$

Since $\mathit{false} \wedge (\cdot)$ is trivially false, by $(\vee L)$ this reduces to:

$$Y : \phi \wedge (\phi\,\mathbf{suf}\,\psi) \Rightarrow y \diamond Y : \phi\,\mathbf{suf}\,\psi$$

By $(\mathbf{suf})$ on the right side and $(\vee R)$, we need to show $Y : \psi$ or $Y : \phi \wedge (\phi\,\mathbf{suf}\,\psi)$. The latter is exactly the left side, so this is immediately closed by (ax).

□

Proposition A.7 (Rule (vii)). *Rule (vii):* $\phi\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi)) \leftrightarrow \mathbf{n}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$ *is derived in G3PPL.*

Proof. Let $X \in \mathit{TVar}$. Recall $\mathbf{n}(\phi \wedge (\phi\,\mathbf{suf}\,\psi)) =_{df} \mathit{false}\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$. Without loss of generality, let $X = y \diamond Y$. We need to prove

$(vii.a)$ $X : \phi\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi)) \Rightarrow X : \mathbf{n}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$.
$(vii.b)$ $X : \mathbf{n}(\phi \wedge (\phi\,\mathbf{suf}\,\psi)) \Rightarrow X : \phi\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$.

Case for proving $(vii.a)$: By $(\mathbf{suf})$ on the left:

$$Y : (\phi \wedge (\phi\,\mathbf{suf}\,\psi)) \vee (\phi \wedge \phi\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))) \Rightarrow y \diamond Y : \mathit{false}\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$$

By $(\mathbf{suf})$ on the right, we get $Y : (\phi \wedge (\phi\,\mathbf{suf}\,\psi))$ and $Y : \mathit{false} \wedge \mathit{false}\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$. Since $\mathit{false} \wedge (\cdot)$ is trivially false, the right side effectively reduces to $Y : \phi \wedge (\phi\,\mathbf{suf}\,\psi)$.

By $(\vee L)$ on the left, we split into 2 cases:

(vii.a1) $Y : \phi \wedge (\phi\,\mathbf{suf}\,\psi) \Rightarrow Y : \phi \wedge (\phi\,\mathbf{suf}\,\psi)$
(vii.a2) $Y : \phi \wedge \phi\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi)) \Rightarrow Y : \phi \wedge (\phi\,\mathbf{suf}\,\psi)$

Case (vii.a1) is immediately closed by (ax).

For case (vii.a2), by $(\wedge L)$ on the left and $(\wedge R)$ on the right, we need to show $Y : \phi$ and $Y : \phi\,\mathbf{suf}\,\psi$ from $Y : \phi$ and $Y : \phi\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$. The branch for $Y : \phi$ on the right is trivially obtained from $Y : \phi$ on the left. For $Y : \phi\,\mathbf{suf}\,\psi$, let $Y = z \diamond Z$. By $(\mathbf{suf})$ on $Y : \phi\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$:

$$Z : (\phi \wedge (\phi\,\mathbf{suf}\,\psi)) \vee (\phi \wedge \phi\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))) \Rightarrow z \diamond Z : \phi\,\mathbf{suf}\,\psi$$

By $(\vee L)$:

(vii.a21) $Z : \phi \wedge (\phi \,\mathbf{suf}\, \psi) \Rightarrow z \diamond Z : \phi \,\mathbf{suf}\, \psi$
(vii.a22) $Z : \phi \wedge \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi)) \Rightarrow z \diamond Z : \phi \,\mathbf{suf}\, \psi$

Case (vii.a21): By $(\wedge L)$, we have $Z : \phi, Z : \phi \,\mathbf{suf}\, \psi \Rightarrow z \diamond Z : \phi \,\mathbf{suf}\, \psi$. By $(\mathbf{suf})$ on $z \diamond Z : \phi \,\mathbf{suf}\, \psi$ on the right, we get $Z : \psi$ and $Z : \phi \wedge (\phi \,\mathbf{suf}\, \psi)$. By $(\wedge R)$, the latter can be proved from the left. This case is trivially closed.

Case (vii.a22): By extending $z \diamond Z : \phi \,\mathbf{suf}\, \psi$ by $(\mathbf{suf})$ on the right, and then weakening, we obtain:

$$(vii.a221)\;\; Z : \phi \wedge \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi)) \Rightarrow Z : \phi \wedge \phi \,\mathbf{suf}\, \psi$$

which has the same form as (ivv.a2). Note that the derivation path from (vii.a2) to (vii.a221) has applied $(\mathbf{suf})$, forming a back-link. The derivation path is progressive since rule $(\mathbf{suf})$ is applied.

Case for proving $(vii.b)$: By $(\mathbf{suf})$ on the left:

$$Y : (\phi \wedge (\phi \,\mathbf{suf}\, \psi)) \vee (\mathit{false} \wedge \mathit{false} \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi))) \Rightarrow y \diamond Y : \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi))$$

Since $\mathit{false} \wedge (\cdot)$ is trivially false, by $(\vee L)$ this simplifies to:

$$(vii.b1)\;\; Y : \phi \wedge (\phi \,\mathbf{suf}\, \psi) \Rightarrow y \diamond Y : \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi))$$

By $(\mathbf{suf})$ on the right side, we get $Y : (\phi \wedge (\phi \,\mathbf{suf}\, \psi))$ and $Y : \phi \wedge \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi))$. By $(\vee R)$, we only need to show one disjunct. The first disjunct $Y : \phi \wedge (\phi \,\mathbf{suf}\, \psi)$ is exactly the left side. This case is immediately closed by (ax).

□

Proposition A.8 (Rule (viii)). *Rule (viii):* $\phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi)) \leftrightarrow \phi \,\mathbf{suf}(\phi \wedge \mathbf{n}\, \psi)$ *is derived in G3PPL.*

Proof. Let $X \in \mathit{TVar}$. Recall that $\mathbf{n}\, \psi =_{df} \mathit{false} \,\mathbf{suf}\, \psi$. We need to prove

$(viii.a)$ $X : \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi)) \Rightarrow X : \phi \,\mathbf{suf}(\phi \wedge \mathbf{n}\, \psi)$.
$(viii.b)$ $X : \phi \,\mathbf{suf}(\phi \wedge \mathbf{n}\, \psi) \Rightarrow X : \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi))$.

Case for proving $(viii.a)$: That is, $X : \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi)) \Rightarrow X : \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$.

Without loss of generality, let $X = x \diamond Y$. By applying $(\mathbf{suf})$ on the left:

$$Y : (\phi \wedge (\phi \,\mathbf{suf}\, \psi)) \vee (\phi \wedge \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi))) \Rightarrow x \diamond Y : \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$$

By $(\vee L)$ on the left, we split into 2 cases:

(viii.a1) $Y : \phi \wedge (\phi \,\mathbf{suf}\, \psi) \Rightarrow x \diamond Y : \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$
(viii.a2) $Y : \phi \wedge \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi)) \Rightarrow x \diamond Y : \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$

Case (viii.a1): By applying $(\mathbf{suf})$ on $x \diamond Y : \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$ on the right, we obtain:

$$Y : \phi \wedge (\phi \,\mathbf{suf}\, \psi) \Rightarrow Y : (\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi), Y : \phi \wedge \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$$

By applying $(\wedge R)$ on $Y : (\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$ on the right:

(viii.a11) $Y : \phi \wedge (\phi \,\mathbf{suf}\, \psi) \Rightarrow Y : \phi, Y : \phi \wedge \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$, which is trivially closed (by $(\wedge L)$ and (ax) on $Y : \phi$).
(viii.a12) $Y : \phi \wedge (\phi \,\mathbf{suf}\, \psi) \Rightarrow Y : \mathit{false} \,\mathbf{suf}\, \psi, Y : \phi \wedge \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$

For case (viii.a12), by applying $(\wedge R)$ on $Y : \phi \wedge \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$ on the right:

(viii.a13) $Y : \phi \wedge (\phi \,\mathbf{suf}\, \psi) \Rightarrow Y : \mathit{false} \,\mathbf{suf}\, \psi, Y : \phi$, which is trivially closed (by $(\wedge L)$ and (ax) on $Y : \phi$).
(viii.a14) $Y : \phi \wedge (\phi \,\mathbf{suf}\, \psi) \Rightarrow Y : \mathit{false} \,\mathbf{suf}\, \psi, Y : \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$

For case (viii.a14), by $(\wedge L)$ on the left, we have $Y : \phi, Y : \phi \,\mathbf{suf}\, \psi$. Rename $Y = y \diamond Z$. By $(\mathbf{suf})$ on $y \diamond Z : \mathit{false} \,\mathbf{suf}\, \psi$ on the right, we get $Z : \psi$ and $Z : \mathit{false} \wedge \mathit{false} \,\mathbf{suf}\, \psi$; since $\mathit{false} \wedge (\cdot)$ is trivially false, the right side effectively contributes only $Z : \psi$. So we need to show:

$$y \diamond Z : \phi, y \diamond Z : \phi \,\mathbf{suf}\, \psi \Rightarrow Z : \psi, y \diamond Z : \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$$

By $(\mathbf{suf})$ on $y \diamond Z : \phi \,\mathbf{suf}\, \psi$ on the left, we obtain:

$$Z : \psi \vee (\phi \wedge \phi \,\mathbf{suf}\, \psi) \Rightarrow Z : \psi, y \diamond Z : \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$$

By $(\vee L)$ on the left, we split into 2 sub-cases:

(viii.a141) $Z : \psi \Rightarrow Z : \psi, y \diamond Z : \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$, which is closed by (ax).
(viii.a142) $Z : \phi \wedge \phi \,\mathbf{suf}\, \psi \Rightarrow Z : \psi, y \diamond Z : \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$

For case (viii.a142), by weakening on $Z : \psi$, it suffices to show:

$$Z : \phi \wedge (\phi \,\mathbf{suf}\, \psi) \Rightarrow y \diamond Z : \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$$

which is exactly (viii.a1) (with $Z, y \diamond Z$ in place of $Y, x \diamond Y$), forming a back-link. The derivation path from (viii.a1) to this back-link is progressive since rule $(\mathbf{suf})$ has been applied (on $y \diamond Z : \phi \,\mathbf{suf}\, \psi$ and on $y \diamond Z : \mathit{false} \,\mathbf{suf}\, \psi$).

Case (viii.a2): By applying $(\mathbf{suf})$ on $x \diamond Y : \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$ on the right, we obtain:

$$Y : \phi \wedge \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi)) \Rightarrow Y : (\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi), Y : \phi \wedge \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$$

By applying $(\wedge R)$ on $Y : (\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$ on the right:

(viii.a21) $Y : \phi \wedge \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi)) \Rightarrow Y : \phi, Y : \phi \wedge \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$, which is trivially closed (by $(\wedge L)$ and (ax) on $Y : \phi$).
(viii.a22) $Y : \phi \wedge \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi)) \Rightarrow Y : \mathit{false} \,\mathbf{suf}\, \psi, Y : \phi \wedge \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$

For case (viii.a22), by applying $(\wedge R)$ on $Y : \phi \wedge \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$ on the right:

(viii.a23) $Y : \phi \wedge \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi)) \Rightarrow Y : \mathit{false} \,\mathbf{suf}\, \psi, Y : \phi, \ldots$, which is trivially closed (by $(\wedge L)$ and (ax) on $Y : \phi$).
(viii.a24) $Y : \phi \wedge \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi)) \Rightarrow Y : \mathit{false} \,\mathbf{suf}\, \psi, Y : \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$

For case (viii.a24), by $(\wedge L)$ and weakening on the left, we obtain:

$$(\mathit{viii.a}241)\quad Y : \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi)) \Rightarrow Y : \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)$$

which is exactly (viii.a), forming a back-link. The derivation path from (viii.a) to (viii.a241) is progressive since rule $(\mathbf{suf})$ is applied.

Case for proving $(\mathit{viii.b})$: That is, $X : \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi) \Rightarrow X : \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi))$.

Without loss of generality, let $X = x \diamond Y$. By applying $(\mathbf{suf})$ on the left:

$$Y : (\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi) \vee (\phi \wedge \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi)) \Rightarrow x \diamond Y : \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi))$$

By $(\vee L)$ on the left, we split into 2 cases:

(viii.b1) $Y : \phi \wedge \mathit{false} \,\mathbf{suf}\, \psi \Rightarrow x \diamond Y : \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi))$
(viii.b2) $Y : \phi \wedge \phi \,\mathbf{suf}(\phi \wedge \mathit{false} \,\mathbf{suf}\, \psi) \Rightarrow x \diamond Y : \phi \,\mathbf{suf}(\phi \wedge (\phi \,\mathbf{suf}\, \psi))$

Case (viii.b1): By applying (**suf**) on $x \diamond Y : \phi\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$ on the right, we obtain:

$$Y : \phi \wedge \mathit{false}\,\mathbf{suf}\,\psi \Rightarrow Y : (\phi \wedge (\phi\,\mathbf{suf}\,\psi)), Y : \phi \wedge \phi\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$$

By applying $(\wedge R)$ on $Y : (\phi \wedge (\phi\,\mathbf{suf}\,\psi))$ on the right:

(viii.b11) $Y : \phi \wedge \mathit{false}\,\mathbf{suf}\,\psi \Rightarrow Y : \phi, Y : \phi \wedge \phi\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$, which is trivially closed (by $(\wedge L)$ and (ax) on $Y : \phi$).
(viii.b12) $Y : \phi \wedge \mathit{false}\,\mathbf{suf}\,\psi \Rightarrow Y : (\phi\,\mathbf{suf}\,\psi), Y : \phi \wedge \phi\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$

For case (viii.b12), rename $Y = y \diamond Z$. By $(\wedge L)$ on the left, we have $y \diamond Z : \phi, y \diamond Z : \mathit{false}\,\mathbf{suf}\,\psi$. By (**suf**) on $y \diamond Z : \mathit{false}\,\mathbf{suf}\,\psi$ on the left, since $\mathit{false} \wedge (\cdot)$ is trivially false, we effectively get $Z : \psi$. By (**suf**) on $y \diamond Z : (\phi\,\mathbf{suf}\,\psi)$ on the right, we get $Z : \psi$ and $Z : \phi \wedge (\phi\,\mathbf{suf}\,\psi)$. Together we need:

$$y \diamond Z : \phi, Z : \psi \Rightarrow Z : \psi, Z : \phi \wedge (\phi\,\mathbf{suf}\,\psi), y \diamond Z : \phi \wedge \phi\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$$

which is trivially closed by (ax) on $Z : \psi$.

Case (viii.b2): By applying (**suf**) on $x \diamond Y : \phi\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$ on the right, we obtain:

$$Y : \phi \wedge \phi\,\mathbf{suf}(\phi \wedge \mathit{false}\,\mathbf{suf}\,\psi) \Rightarrow Y : (\phi \wedge (\phi\,\mathbf{suf}\,\psi)), Y : \phi \wedge \phi\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$$

By weakening on $Y : (\phi \wedge (\phi\,\mathbf{suf}\,\psi))$, it suffices to show:

$$Y : \phi \wedge \phi\,\mathbf{suf}(\phi \wedge \mathit{false}\,\mathbf{suf}\,\psi) \Rightarrow Y : \phi \wedge \phi\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$$

By applying $(\wedge R)$ on the right:

(viii.b21) $Y : \phi \wedge \phi\,\mathbf{suf}(\phi \wedge \mathit{false}\,\mathbf{suf}\,\psi) \Rightarrow Y : \phi$, which is trivially closed (by $(\wedge L)$ and (ax) on $Y : \phi$).
(viii.b22) $Y : \phi \wedge \phi\,\mathbf{suf}(\phi \wedge \mathit{false}\,\mathbf{suf}\,\psi) \Rightarrow Y : \phi\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$

For case (viii.b22), by $(\wedge L)$ and weakening on the left, we obtain:

$$(viii.b221)\quad Y : \phi\,\mathbf{suf}(\phi \wedge \mathit{false}\,\mathbf{suf}\,\psi) \Rightarrow Y : \phi\,\mathbf{suf}(\phi \wedge (\phi\,\mathbf{suf}\,\psi))$$

which is exactly (viii.b), forming a back-link. The derivation path from (viii.b) to (viii.b221) is progressive since rule (**suf**) is applied.

□

Proposition A.9 (Rule (ix)). *Rule (ix):* $\mathbf{f}\,\mathbf{n}\,\mathit{true} \leftrightarrow \mathit{false}$ *is derived in G3PPL.*

Proof. Let $X \in \mathit{TVar}$. Recall that $\mathbf{n}\,\mathit{true} =_{df} \mathit{false}\,\mathbf{suf}\,\mathit{true}$. We need to prove

$(ix.a)$ $X : \mathbf{f}(\mathit{false}\,\mathbf{suf}\,\mathit{true}) \Rightarrow X : \mathit{false}$.
$(ix.b)$ $X : \mathit{false} \Rightarrow X : \mathbf{f}(\mathit{false}\,\mathbf{suf}\,\mathit{true})$.

Case for proving $(ix.a)$: Without loss of generality, let $X = x \diamond Y$. By the rule $(\mathbf{f})$ on $x \diamond Y : \mathbf{f}(\mathit{false}\,\mathbf{suf}\,\mathit{true})$:

$$x : \mathit{false}\,\mathbf{suf}\,\mathit{true} \Rightarrow x \diamond Y : \mathit{false}$$

By (**suf** x), the sequent is closed.

Case for proving $(ix.b)$: This is trivially closed since $X : \mathit{false}$ is on the left side.

□

Proposition A.10 (Rule (x)). *Rule (x):* $\neg\phi \wedge \mathbf{f}\,\phi \rightarrow \mathbf{n}\,\mathit{true}$ *is derived in G3PPL.*

Proof. Let $X \in \mathit{TVar}$. We need to prove

$$X : \neg\phi \wedge \mathbf{f}\,\phi \Rightarrow X : \mathbf{n}\ \mathit{true}.$$

That is, $X : \neg\phi \wedge \mathbf{f}\,\phi \Rightarrow X : \mathit{false}\ \mathbf{suf}\ \mathit{true}$.

Without loss of generality, let $X = x \diamond Y$. By $(\wedge L)$ on the left:

$$x \diamond Y : \neg\phi, x \diamond Y : \mathbf{f}\,\phi \Rightarrow x \diamond Y : \mathit{false}\ \mathbf{suf}\ \mathit{true}$$

By the rule $(\mathbf{f})$ on $x \diamond Y : \mathbf{f}\,\phi$:

$$x \diamond Y : \neg\phi, x : \phi \Rightarrow x \diamond Y : \mathit{false}\ \mathbf{suf}\ \mathit{true}$$

By $(\mathbf{suf})$ on $x \diamond Y : \mathit{false}\ \mathbf{suf}\ \mathit{true}$ on the right side, we obtain:

$$x \diamond Y : \neg\phi, x : \phi \Rightarrow Y : \mathit{true} \vee (\mathit{false} \wedge \mathit{false}\ \mathbf{suf}\ \mathit{true})$$

Since $\mathit{false} \wedge (\cdot)$ is trivially false, by $(\vee R)$ it suffices to show:

$$x \diamond Y : \neg\phi, x : \phi \Rightarrow Y : \mathit{true}$$

which is trivially closed since $Y : \mathit{true}$ is always provable.

□

Proposition A.11 (Rule (xi)). *Rule (xi): $p \leftrightarrow \mathbf{f}\,p$, where p is an atomic proposition, is derived in G3PPL.*

Proof. Let $X \in \mathit{TVar}$. We need to prove

$(xi.a)$ $X : p \Rightarrow X : \mathbf{f}\,p$.

$(xi.b)$ $X : \mathbf{f}\,p \Rightarrow X : p$.

Case for proving $(xi.a)$: Without loss of generality, let $X = x \diamond Y$. By the rule $(\mathbf{f})$ on $x \diamond Y : \mathbf{f}\,p$ on the right:

$$x \diamond Y : p \Rightarrow x : p$$

By the rule (p) on $x \diamond Y : p$ on the left:

$$x : p \Rightarrow x : p$$

which is immediately closed by (ax).

Case for proving $(xi.b)$: Without loss of generality, let $X = x \diamond Y$. By the rule $(\mathbf{f})$ on $x \diamond Y : \mathbf{f}\,p$ on the left:

$$x : p \Rightarrow x \diamond Y : p$$

By the rule (p) on $x \diamond Y : p$ on the right:

$$x : p \Rightarrow x : p$$

which is immediately closed by (ax). □

Proposition A.12 (Rule (xii)). *Rule (xii): $\mathbf{f}\,\phi \wedge \langle\alpha\rangle\psi \leftrightarrow \langle\alpha\rangle(\mathbf{f}\,\phi \wedge \psi)$ is derived in G3PPL.*

Proof. Let $X \in \mathit{TVar}$. We prove both following directions by structural induction on α. Throughout, recall that $\langle\alpha\rangle\theta =_{df} \neg[\alpha]\neg\theta$.

$(xii.a)$ $X : \mathbf{f}\,\phi \wedge \langle\alpha\rangle\psi \Rightarrow X : \langle\alpha\rangle(\mathbf{f}\,\phi \wedge \psi)$.

$(xii.b)$ $X : \langle\alpha\rangle(\mathbf{f}\,\phi \wedge \psi) \Rightarrow X : \mathbf{f}\,\phi \wedge \langle\alpha\rangle\psi$.

Case for proving $(xii.a)$: By $(\wedge L)$ on the left, we need to show:

$$X : \mathbf{f}\,\phi, X : \langle\alpha\rangle\psi \Rightarrow X : \langle\alpha\rangle(\mathbf{f}\,\phi \wedge \psi)$$

Case $\alpha = a$ **(atomic action).** Let $X = Y \diamond x$. We need:

$$(xii.a.a)\quad Y \diamond x : \mathbf{f}\,\phi, Y \diamond x : \langle a\rangle\psi \Rightarrow Y \diamond x : \langle a\rangle(\mathbf{f}\,\phi \wedge \psi)$$

By $(\langle a\rangle R)$ and $(\langle a\rangle L)$, it suffices to show, for a new y:

$$Y \diamond x : \mathbf{f}\,\phi, Y \diamond x \diamond y : \psi \Rightarrow Y \diamond x \diamond y : \mathbf{f}\,\phi \wedge \psi$$

By $(\wedge R)$ on the right, we need $Y \diamond x \diamond y : \mathbf{f}\,\phi$ and $Y \diamond x \diamond y : \psi$. The latter is on the left. For the former, let $Y = z \diamond Z$, by $(\mathbf{f})$ on $Y \diamond x : \mathbf{f}\,\phi$ on the left we get $z : \phi$, and by $(\mathbf{f})$ on $Y \diamond x \diamond y : \mathbf{f}\,\phi$ on the right we need $z : \phi$, which is available. This case is closed.

Case $\alpha = \chi?$ **(test).** Let $X = Y \diamond x$. We need:

$$(xii.a.\chi?)\quad Y \diamond x : \mathbf{f}\,\phi, Y \diamond x : \langle\chi?\rangle\psi \Rightarrow Y \diamond x : \langle\chi?\rangle(\mathbf{f}\,\phi \wedge \psi)$$

By $(\langle\phi?\rangle L)$ on the left, we get $x \diamond x : \chi$ and $Y \diamond x \diamond x : \psi$. By $(\langle\phi?\rangle R)$ on the right, we need:

(xii.a.χ?.1) $Y \diamond x : \mathbf{f}\,\phi, x \diamond x : \chi, Y \diamond x \diamond x : \psi \Rightarrow x \diamond x : \chi$, which is closed by (ax).

(xii.a.χ?.2) $Y \diamond x : \mathbf{f}\,\phi, x \diamond x : \chi, Y \diamond x \diamond x : \psi \Rightarrow Y \diamond x \diamond x : (\mathbf{f}\,\phi \wedge \psi)$. By $(\wedge R)$, we need $Y \diamond x \diamond x : \mathbf{f}\,\phi$ and $Y \diamond x \diamond x : \psi$. The latter is on the left. For the former, let $Y = z \diamond Z$, by $(\mathbf{f})$ on $Y \diamond x : \mathbf{f}\,\phi$ on the left we get $z : \phi$, and by $(\mathbf{f})$ on $Y \diamond x \diamond x : \mathbf{f}\,\phi$ on the right we need $z : \phi$, which is available. This case is closed.

Case $\alpha = \alpha_1 ; \alpha_2$ **(sequential composition).** We need to prove:

$$(xii.a.;)\quad X : \mathbf{f}\,\phi, X : \langle\alpha_1 ; \alpha_2\rangle\psi \Rightarrow X : \langle\alpha_1 ; \alpha_2\rangle(\mathbf{f}\,\phi \wedge \psi)$$

By $\langle;\rangle$, we need:

$$(xii.a.;.1)\quad X : \mathbf{f}\,\phi, X : \langle\alpha_1\rangle\langle\alpha_2\rangle\psi \Rightarrow X : \langle\alpha_1\rangle\langle\alpha_2\rangle(\mathbf{f}\,\phi \wedge \psi)$$

By the induction hypothesis for α_1:

$$(xii.;.11)\quad X : \mathbf{f}\,\phi, X : \langle\alpha_1\rangle\langle\alpha_2\rangle\psi \Rightarrow X : \langle\alpha_1\rangle(\mathbf{f}\,\phi \wedge \langle\alpha_2\rangle\psi)$$

is provable. So from (xii.;.1), by applying (cut), we need to prove (xii.;.11) and :

$$(xii.;.12)\quad X : \langle\alpha_1\rangle(\mathbf{f}\,\phi \wedge \langle\alpha_2\rangle\psi) \Rightarrow X : \langle\alpha_1\rangle\langle\alpha_2\rangle(\mathbf{f}\,\phi \wedge \psi)$$

By the Necessitation Lemma we then need:

$$X : (\mathbf{f}\,\phi \wedge \langle\alpha_2\rangle\psi) \Rightarrow X : \langle\alpha_2\rangle(\mathbf{f}\,\phi \wedge \psi)$$

But this is straightforward by induction hypothesis.

Case $\alpha = \alpha_1 \cup \alpha_2$ **(choice).** We need to prove

$$(xii.a.\cup)\quad X : \mathbf{f}\,\phi, X : \langle\alpha_1\rangle\psi \vee \langle\alpha_2\rangle\psi \Rightarrow X : \langle\alpha_1\rangle(\mathbf{f}\,\phi \wedge \psi) \vee \langle\alpha_2\rangle(\mathbf{f}\,\phi \wedge \psi)$$

By $(\vee L)$ and $(\vee R)$, we need to prove both:

(xii.a.$\cup$.1) $X : \mathbf{f}\,\phi, X : \langle\alpha_1\rangle\psi \Rightarrow X : \langle\alpha_1\rangle(\mathbf{f}\,\phi \wedge \psi), X : \langle\alpha_2\rangle(\mathbf{f}\,\phi \wedge \psi)$

(xii.a.$\cup$.1) $X : \mathbf{f}\ \ phi, X : \langle\alpha_2\rangle\psi \Rightarrow X : \langle\alpha_1\rangle(\mathbf{f}\,\phi \wedge \psi), X : \langle\alpha_2\rangle(\mathbf{f}\,\phi \wedge \psi)$

Both are direct by weakening and induction hypothesis. For example, from (xii.a.∪.1), by (wk), we need:

$$X : \mathbf{f}\,\phi, X : \langle\alpha_1\rangle\psi \Rightarrow X : \langle\alpha_1\rangle(\mathbf{f}\,\phi \wedge \psi),$$

which is provable by induction hypothesis.

Case $\alpha = \alpha_1^*$ **(iteration).** We need to prove:

$$(xii.a.*)\;\; X : \mathbf{f}\,\phi, X : \langle\alpha_1^*\rangle\psi \Rightarrow X : \langle\alpha_1^*\rangle(\mathbf{f}\,\phi \wedge \psi)$$

By $\langle * \rangle$, $X : \langle\alpha_1^*\rangle\psi$ gives $X : \langle \mathit{true}?\cup\alpha_1\,;\alpha_1^*\rangle\psi$, i.e., $X : \langle \mathit{true}?\rangle\psi$ or $X : \langle\alpha_1\rangle\langle\alpha_1^*\rangle\psi$. So, we split (xii.a.∗) into:

(xii.a.∗.1) $X : \langle \mathit{true}?\rangle\psi \Rightarrow X : \langle\alpha_1^*\rangle(\mathbf{f}\,\phi \wedge \psi)$
(xii.a.∗.2) $X : \langle\alpha_1\rangle\langle\alpha_1^*\rangle\psi \Rightarrow X : \langle\alpha_1^*\rangle(\mathbf{f}\,\phi \wedge \psi)$

Case (xii.a.∗.1): By $\langle * \rangle$ on the right, $X : \langle\alpha_1^*\rangle(\mathbf{f}\,\phi \wedge \psi)$ gives $X : \langle \mathit{true}?\cup\alpha_1\,;\alpha_1^*\rangle(\mathbf{f}\,\phi \wedge \psi)$. It suffices to show

$$X : \langle \mathit{true}?\rangle\psi \Rightarrow X : \langle \mathit{true}?\rangle(\mathbf{f}\,\phi \wedge \psi).$$

This is straightforward. Let $X = Y \diamond x$, then by rule $(\chi?)$ on the right we have

$$X : \langle \mathit{true}?\rangle\psi \Rightarrow x \diamond x : \mathit{true}, Y \diamond x \diamond x : (\mathbf{f}\,\phi \wedge \psi).$$

This closes since we have $x \diamond x : \mathit{true}$ on the right.

Case (xii.a.∗.2): By $\langle * \rangle$ on the right, it suffices to show

$$(xii.a.*.21)\;\; X : \langle\alpha_1\rangle\langle\alpha_1^*\rangle\psi \Rightarrow X : \langle\alpha_1\rangle\langle\alpha_1^*\rangle(\mathbf{f}\,\phi \wedge \psi).$$

By the induction hypothesis for α_1 (with $\langle\alpha_1^*\rangle\psi$ in place of ψ), we know that

$$(xii.a.*.211)\;\; X : \mathbf{f}\,\phi, X : \langle\alpha_1\rangle\langle\alpha_1^*\rangle\psi \Rightarrow X : \langle\alpha_1\rangle(\mathbf{f}\,\phi \wedge \langle\alpha_1^*\rangle\psi)$$

is provable. So, from (xii.a.∗.21), by (cut), we need to prove (xii.a.*.211) and:

$$(xii.a.*.212)\;\; X : \langle\alpha_1\rangle(\mathbf{f}\,\phi \wedge \langle\alpha_1^*\rangle\psi) \Rightarrow X : \langle\alpha_1\rangle\langle\alpha_1^*\rangle(\mathbf{f}\,\phi \wedge \psi)$$

From (xii.a.∗.212), by Necessitation Lemma, we obtain:

$$(xii.a.*.2121)\;\; X : (\mathbf{f}\,\phi \wedge \langle\alpha_1^*\rangle\psi) \Rightarrow X : \langle\alpha_1^*\rangle(\mathbf{f}\,\phi \wedge \psi)$$

which is exactly (xii.a.∗) (after applying rule $(\wedge L)$). And it is a back-link since from (xii.a.∗) to (xii.a.*.2121), Necessitation Lemma has been applied.

Case for proving $(xii.b)$:

Case $\alpha = a$ **(atomic action).** Let $X = Y \diamond x$. We need:

$$(xii.b.a)\;\; Y \diamond x : \langle a\rangle(\mathbf{f}\,\phi \wedge \psi) \Rightarrow Y \diamond x : \mathbf{f}\,\phi \wedge \langle a\rangle\psi$$

By $(\wedge R)$ on the right, we split into:

(xii.b.a.1) $Y \diamond x : \langle a\rangle(\mathbf{f}\,\phi \wedge \psi) \Rightarrow Y \diamond x : \mathbf{f}\,\phi$.
By $(\langle\alpha\rangle)L$, it suffices to show $x\,\mathrm{R}_a\,y, Y \diamond x \diamond y : (\mathbf{f}\,\phi \wedge \psi) \Rightarrow Y \diamond x : \mathbf{f}\,\phi$ with a new y. By $(\wedge L)$, we have $Y \diamond x \diamond y : \mathbf{f}\,\phi$. Let $Y = z \diamond Z$. By $(\mathbf{f})$ on $Y \diamond x \diamond y : \mathbf{f}\,\phi$ on the left, we get $z : \phi$. By $(\mathbf{f})$ on $Y \diamond x : \mathbf{f}\,\phi$ on the right, we need $z : \phi$, which is available. This case is closed.

(xii.b.a.2) $Y \diamond x : \langle a\rangle(\mathbf{f}\,\phi \wedge \psi) \Rightarrow Y \diamond x : \langle a\rangle\psi$.

This follows from the Necessitation Lemma applied to $X : \mathbf{f}\,\phi \wedge \psi \Rightarrow X : \psi$ (trivially derivable by $(\wedge L)$ and (ax)).

Case $\alpha = \chi?$ **(test).** Let $X = Y \diamond x$. We need:

$$(xii.b.\chi?)\quad Y \diamond x : \langle \chi? \rangle (\mathbf{f}\,\phi \wedge \psi) \Rightarrow Y \diamond x : \mathbf{f}\,\phi \wedge \langle \chi? \rangle \psi$$

By $(\langle \phi? \rangle L)$ on the left, we get $x \diamond x : \chi$ and $Y \diamond x \diamond x : \mathbf{f}\,\phi \wedge \psi$. By $(\wedge R)$ on the right:

(xii.b.χ?.1) $x \diamond x : \chi, Y \diamond x \diamond x : \mathbf{f}\,\phi \wedge \psi \Rightarrow Y \diamond x : \mathbf{f}\,\phi$.
By $(\wedge L)$, we have $Y \diamond x \diamond x : \mathbf{f}\,\phi$. Let $Y = z \diamond Z$. By $(\mathbf{f})$ on $Y \diamond x \diamond x : \mathbf{f}\,\phi$ on the left, we get $z : \phi$. By $(\mathbf{f})$ on $Y \diamond x : \mathbf{f}\,\phi$ on the right, we need $z : \phi$, which is available. This case is closed.

(xii.b.χ?.2) $x \diamond x : \chi, Y \diamond x \diamond x : \mathbf{f}\,\phi \wedge \psi \Rightarrow Y \diamond x : \langle \chi? \rangle \psi$.
By $(\langle \phi? \rangle R)$ on the right, we need $x \diamond x : \chi$ (available on the left) and $Y \diamond x \diamond x : \psi$ (obtained from $Y \diamond x \diamond x : \mathbf{f}\,\phi \wedge \psi$ by $(\wedge L)$ and (ax)). This case is closed.

Case $\alpha = \alpha_1 ; \alpha_2$ **(sequential composition).** We need:

$$(xii.b.;)\quad X : \langle \alpha_1 ; \alpha_2 \rangle (\mathbf{f}\,\phi \wedge \psi) \Rightarrow X : \mathbf{f}\,\phi \wedge \langle \alpha_1 ; \alpha_2 \rangle \psi$$

By $\langle ; \rangle$, this becomes:

$$X : \langle \alpha_1 \rangle \langle \alpha_2 \rangle (\mathbf{f}\,\phi \wedge \psi) \Rightarrow X : \mathbf{f}\,\phi \wedge \langle \alpha_1 \rangle \langle \alpha_2 \rangle \psi$$

By the induction hypothesis for α_2, $X : \langle \alpha_2 \rangle (\mathbf{f}\,\phi \wedge \psi) \Rightarrow X : \mathbf{f}\,\phi \wedge \langle \alpha_2 \rangle \psi$ is provable. By the Necessitation Lemma:

$$(xii.b.;.1)\quad X : \langle \alpha_1 \rangle \langle \alpha_2 \rangle (\mathbf{f}\,\phi \wedge \psi) \Rightarrow X : \langle \alpha_1 \rangle (\mathbf{f}\,\phi \wedge \langle \alpha_2 \rangle \psi)$$

is provable. By the induction hypothesis for α_1 (with $\langle \alpha_2 \rangle \psi$ in place of ψ):

$$(xii.b.;.2)\quad X : \langle \alpha_1 \rangle (\mathbf{f}\,\phi \wedge \langle \alpha_2 \rangle \psi) \Rightarrow X : \mathbf{f}\,\phi \wedge \langle \alpha_1 \rangle \langle \alpha_2 \rangle \psi$$

is provable. The result follows by (cut) on (xii.b.;.1) and (xii.b.;.2).

Case $\alpha = \alpha_1 \cup \alpha_2$ **(choice).** We need:

$$(xii.b.\cup)\quad X : \langle \alpha_1 \cup \alpha_2 \rangle (\mathbf{f}\,\phi \wedge \psi) \Rightarrow X : \mathbf{f}\,\phi \wedge \langle \alpha_1 \cup \alpha_2 \rangle \psi$$

By $\langle \cup \rangle$, $X : \langle \alpha_1 \cup \alpha_2 \rangle (\mathbf{f}\,\phi \wedge \psi)$ gives $X : \langle \alpha_1 \rangle (\mathbf{f}\,\phi \wedge \psi)$ or $X : \langle \alpha_2 \rangle (\mathbf{f}\,\phi \wedge \psi)$. Consider the sub-case $X : \langle \alpha_1 \rangle (\mathbf{f}\,\phi \wedge \psi)$: by the induction hypothesis for α_1:

$$X : \langle \alpha_1 \rangle (\mathbf{f}\,\phi \wedge \psi) \Rightarrow X : \mathbf{f}\,\phi \wedge \langle \alpha_1 \rangle \psi$$

is provable. Since $\langle \alpha_1 \rangle \psi$ implies $\langle \alpha_1 \cup \alpha_2 \rangle \psi$ (by $([\cup])$), we obtain $X : \mathbf{f}\,\phi \wedge \langle \alpha_1 \cup \alpha_2 \rangle \psi$ by (cut). The sub-case for α_2 is symmetric.

Case $\alpha = \alpha_1^*$ **(iteration).** We need:

$$(xii.b.*)\quad X : \langle \alpha_1^* \rangle (\mathbf{f}\,\phi \wedge \psi) \Rightarrow X : \mathbf{f}\,\phi \wedge \langle \alpha_1^* \rangle \psi$$

By $(\langle * \rangle)$, $X : \langle \alpha_1^* \rangle (\mathbf{f}\,\phi \wedge \psi)$ gives $X : \langle \mathit{true}? \rangle (\mathbf{f}\,\phi \wedge \psi)$ or $X : \langle \alpha_1 \rangle \langle \alpha_1^* \rangle (\mathbf{f}\,\phi \wedge \psi)$. Consider two cases:

(vii.b.$*$.1) $X : \langle \mathit{true}? \rangle (\mathbf{f}\,\phi \wedge \psi) \Rightarrow X : \mathbf{f}\,\phi \wedge \langle \alpha_1^* \rangle \psi$

(vii.b.$*$.2) $X : \langle \alpha_1 \rangle \langle \alpha_1^* \rangle (\mathbf{f}\,\phi \wedge \psi) \Rightarrow X : \mathbf{f}\,\phi \wedge \langle \alpha_1^* \rangle \psi$

For (vii.b.$*$.1), let $X = Y \diamond x$, then by $(\langle \phi? L \rangle)$, $Y : \langle \mathit{true}? \rangle (\mathbf{f}\,\phi \wedge \psi)$ gives $x \diamond x : \mathit{true}, X \diamond x \diamond x : (\mathbf{f}\,\phi \wedge \psi)$. By $(\wedge R)$, we need to prove:

(vii.b.$*$.11) $Y \diamond x \diamond x : (\mathbf{f}\,\phi \wedge \psi) \Rightarrow Y \diamond x : \mathbf{f}\,\phi$

(vii.b.$*$.12) $Y \diamond x \diamond x : (\mathbf{f}\,\phi \wedge \psi) \Rightarrow Y \diamond x : \langle \alpha_1^* \rangle \psi$

(vii.b.$*$.11) can be obtained by applying ($\mathbf{f}$) on both sides since they both contain $\mathbf{f}\,\phi$. (vii.b.$*$.12) can be obtained by applying ($\langle * \rangle$) on the right side to match $x \diamond x : \mathit{true}$.

For (xii.b.$*$.2), by the induction hypothesis for α_1, we know that

$$(\mathit{xii.b.*.21})\;\; X : \langle \alpha_1 \rangle (\mathbf{f}\,\phi \wedge \langle \alpha_1^* \rangle \psi) \Rightarrow X : \mathbf{f}\,\phi \wedge \langle \alpha_1 \rangle \langle \alpha_1^* \rangle \psi$$

is provable. So by (cut) and that fact that $\langle \alpha_1 \rangle \langle \alpha_1^* \rangle \psi$ actually implies $\langle \alpha_1^* \rangle \psi$, to prove (vii.b.$*$.2) it is sufficient to prove both (xii.b.$*$.21) (which has been proved by the induction hypothesis) and:

$$(\mathit{xii.b.*.22})\;\; X : \langle \alpha_1 \rangle \langle \alpha_1^* \rangle (\mathbf{f}\,\phi \wedge \psi) \Rightarrow X : \langle \alpha_1 \rangle (\mathbf{f}\,\phi \wedge \langle \alpha_1^* \rangle \psi).$$

By Necessitation Lemma, (xii.b.*.22) gives

$$(\mathit{xii.b.*.221})\;\; X : \langle \alpha_1^* \rangle (\mathbf{f}\,\phi \wedge \psi) \Rightarrow X : (\mathbf{f}\,\phi \wedge \langle \alpha_1^* \rangle \psi),$$

which is exactly (xii.b.*). We thus obtain a cyclic derivation from (xii.b.$*$) to (xii.b.$*$.211) since during which the Necessitation Lemma is applied.

□

Proposition A.13 (Rule (xiii)). *Rule (xiii):* $\mathbf{n}\langle \alpha \rangle \phi \leftrightarrow \mathbf{n}\,\mathit{true} \wedge \langle \alpha \rangle\, \bar{\mathbf{n}}\, \phi$ *is derived in G3PPL.*

Proof. Let $X \in \mathit{TVar}$. Recall that $\mathbf{n}\,\theta =_{df} \mathit{false}\,\mathbf{suf}\,\theta$ and $\bar{\mathbf{n}}\,\theta =_{df} \neg\,\mathbf{n}\,\neg\theta = \neg(\mathit{false}\,\mathbf{suf}\,\neg\theta)$. We prove both following directions by structural induction on α.

($\mathit{xiii.a}$) $X : \mathbf{n}\langle \alpha \rangle \phi \Rightarrow X : \mathbf{n}\,\mathit{true} \wedge \langle \alpha \rangle\, \bar{\mathbf{n}}\, \phi$.

($\mathit{xiii.b}$) $X : \mathbf{n}\,\mathit{true} \wedge \langle \alpha \rangle\, \bar{\mathbf{n}}\, \phi \Rightarrow X : \mathbf{n}\langle \alpha \rangle \phi$.

Case for proving ($\mathit{xiii.a}$): That is, $X : \mathit{false}\,\mathbf{suf}\langle \alpha \rangle \phi \Rightarrow X : (\mathit{false}\,\mathbf{suf}\,\mathit{true}) \wedge \langle \alpha \rangle \neg (\mathit{false}\,\mathbf{suf}\,\neg\phi)$.

Without loss of generality, let $X = x \diamond Y$. By ($\mathbf{suf}$) on $x \diamond Y : \mathit{false}\,\mathbf{suf}\langle \alpha \rangle \phi$ on the left, since $\mathit{false} \wedge (\cdot)$ is trivially false, we get:

$$Y : \langle \alpha \rangle \phi \Rightarrow x \diamond Y : (\mathit{false}\,\mathbf{suf}\,\mathit{true}) \wedge \langle \alpha \rangle \neg (\mathit{false}\,\mathbf{suf}\,\neg\phi)$$

By ($\wedge R$) on the right, we split into:

(xiii.a.1) $Y : \langle \alpha \rangle \phi \Rightarrow x \diamond Y : \mathit{false}\,\mathbf{suf}\,\mathit{true}$.
By ($\mathbf{suf}$) on $x \diamond Y : \mathit{false}\,\mathbf{suf}\,\mathit{true}$ on the right, since $\mathit{false} \wedge (\cdot)$ is trivially false, we need $Y : \mathit{true}$, which is always provable.

(xiii.a.2) $Y : \langle \alpha \rangle \phi \Rightarrow x \diamond Y : \langle \alpha \rangle \neg (\mathit{false}\,\mathbf{suf}\,\neg\phi)$

For case (xiii.a.2), we prove by structural induction on α.

Case $\alpha = a$ **(atomic action).** Let $Y = Z \diamond y$. We need:

$$Z \diamond y : \langle a \rangle \phi \Rightarrow x \diamond Z \diamond y : \langle a \rangle \neg (\mathit{false}\,\mathbf{suf}\,\neg\phi)$$

By ($\langle a \rangle L$) on the left, we get $y\,\mathrm{R}_a\,z'$ and $Z \diamond y \diamond z' : \phi$. By ($\langle a \rangle R$) on the right (with the same z'), we need $x \diamond Z \diamond y \diamond z' : \neg(\mathit{false}\,\mathbf{suf}\,\neg\phi)$. By ($\neg R$):

$$Z \diamond y \diamond z' : \phi, x \diamond Z \diamond y \diamond z' : \mathit{false}\,\mathbf{suf}\,\neg\phi \Rightarrow \cdot$$

By ($\mathbf{suf}$) on $x \diamond Z \diamond y \diamond z' : \mathit{false}\,\mathbf{suf}\,\neg\phi$ on the left, since $\mathit{false} \wedge (\cdot)$ is trivially false, we get $Z \diamond y \diamond z' : \neg\phi$. Then $Z \diamond y \diamond z' : \phi$ and $Z \diamond y \diamond z' : \neg\phi$ gives a contradiction, closing by ($\neg L$) and (ax).

Case $\alpha = \chi?$ **(test).** Let $Y = Z \diamond y$. We need:

$$Z \diamond y : \langle \chi? \rangle \phi \Rightarrow x \diamond Z \diamond y : \langle \chi? \rangle \neg (\mathit{false}\, \mathbf{suf}\, \neg \phi)$$

By $(\langle \phi? \rangle L)$ on the left, we get $y \diamond y : \chi$ and $Z \diamond y \diamond y : \phi$. By $(\langle \phi? \rangle R)$ on the right, we need:

(xiii.a.χ?.1) $y \diamond y : \chi, Z \diamond y \diamond y : \phi \Rightarrow y \diamond y : \chi$, closed by (ax).

(xiii.a.χ?.2) $y \diamond y : \chi, Z \diamond y \diamond y : \phi \Rightarrow x \diamond Z \diamond y \diamond y : \neg(\mathit{false}\, \mathbf{suf}\, \neg\phi)$. By $(\neg R)$, we need $Z \diamond y \diamond y : \phi, x \diamond Z \diamond y \diamond y : \mathit{false}\, \mathbf{suf}\, \neg\phi \Rightarrow \cdot$. By $(\mathbf{suf})$, since $\mathit{false} \wedge (\cdot)$ is trivially false, we get $Z \diamond y \diamond y : \neg\phi$. Contradiction with $Z \diamond y \diamond y : \phi$. Closed.

Case $\alpha = \alpha_1 ; \alpha_2$ **(sequential composition).** By $(\langle ; \rangle)$:

$$Y : \langle \alpha_1 \rangle \langle \alpha_2 \rangle \phi \Rightarrow x \diamond Y : \langle \alpha_1 \rangle \langle \alpha_2 \rangle \neg (\mathit{false}\, \mathbf{suf}\, \neg \phi)$$

By the induction hypothesis for α_1 (with $\langle \alpha_2 \rangle \phi$ in place of ϕ):

$$(xiii.a.;.1) \quad Y : \langle \alpha_1 \rangle \langle \alpha_2 \rangle \phi \Rightarrow x \diamond Y : \langle \alpha_1 \rangle \neg (\mathit{false}\, \mathbf{suf}\, \neg \langle \alpha_2 \rangle \phi)$$

is provable. We also need:

$$(xiii.a.;.2) \quad x \diamond Y : \langle \alpha_1 \rangle \neg (\mathit{false}\, \mathbf{suf}\, \neg \langle \alpha_2 \rangle \phi) \Rightarrow x \diamond Y : \langle \alpha_1 \rangle \langle \alpha_2 \rangle \neg (\mathit{false}\, \mathbf{suf}\, \neg \phi)$$

so that the result can follow by (cut). By the Necessitation Lemma, it suffices to show:

$$(xiii.a.;.21) \quad x \diamond Y : \neg (\mathit{false}\, \mathbf{suf}\, \neg \langle \alpha_2 \rangle \phi) \Rightarrow x \diamond Y : \langle \alpha_2 \rangle \neg (\mathit{false}\, \mathbf{suf}\, \neg \phi)$$

By (cut), it is equivalent to show

(xiii.a.;.211) $x \diamond Y : \neg (\mathit{false}\, \mathbf{suf}\, \neg \langle \alpha_2 \rangle \phi) \Rightarrow Y : \langle \alpha_2 \rangle \phi$.

(xiii.a.;.212) $Y : \langle \alpha_2 \rangle \phi \Rightarrow x \diamond Y : \langle \alpha_2 \rangle \neg (\mathit{false}\, \mathbf{suf}\, \neg \phi)$.

(xiii.a.;.211) is not hard to archieve by applying $(\neg R)$ and $(\mathbf{suf})$ on the left to get both $Y : \neg \langle \alpha_2 \rangle \phi$ and $Y : \langle \alpha_2 \rangle \phi$ on the left, which closes the branch.

(xiii.a.;.212) is exactly provable by induction hypothesis on α_2.

Case $\alpha = \alpha_1 \cup \alpha_2$ **(choice).** We need

$$Y : \langle \alpha_1 \cup \alpha_2 \rangle \phi \Rightarrow x \diamond Y : \langle \alpha_1 \cup \alpha_2 \rangle \neg (\mathit{false}\, \mathbf{suf}\, \neg \phi).$$

By $(\langle \cup \rangle)$, $Y : \langle \alpha_1 \cup \alpha_2 \rangle \phi$ gives $Y : \langle \alpha_1 \rangle \phi$ or $Y : \langle \alpha_2 \rangle \phi$. In the sub-case $Y : \langle \alpha_1 \rangle \phi$: by the induction hypothesis for α_1, we get $x \diamond Y : \langle \alpha_1 \rangle \neg (\mathit{false}\, \mathbf{suf}\, \neg \phi)$. Since $Y : \langle \alpha_1 \rangle \theta$ implies $Y : \langle \alpha_1 \cup \alpha_2 \rangle \theta$ (by $(\langle \cup \rangle)$), we obtain $x \diamond Y : \langle \alpha_1 \cup \alpha_2 \rangle \neg (\mathit{false}\, \mathbf{suf}\, \neg \phi)$. The sub-case for α_2 is symmetric.

Case $\alpha = \alpha_1^*$ **(iteration).** We need to prove

$$(xiii.a.*) \quad Y : \langle \alpha_1^* \rangle \phi \Rightarrow x \diamond Y : \langle \alpha_1^* \rangle \neg (\mathit{false}\, \mathbf{suf}\, \neg \phi).$$

By $(\langle * \rangle)$, $Y : \langle \alpha_1^* \rangle \phi$ gives $Y : \langle \mathit{true}? \rangle \phi$ or $Y : \langle \alpha_1 \rangle \langle \alpha_1^* \rangle \phi$. We need

(xiii.a.*.1) $Y : \langle \mathit{true}? \rangle \phi \Rightarrow x \diamond Y : \langle \alpha_1^* \rangle \neg (\mathit{false}\, \mathbf{suf}\, \neg \phi)$.

(xiii.a.*.2) $Y : \langle \alpha_1 \rangle \langle \alpha_1^* \rangle \phi \Rightarrow x \diamond Y : \langle \alpha_1^* \rangle \neg (\mathit{false}\, \mathbf{suf}\, \neg \phi)$

For (xiii.a.*.1): It gives $Z \diamond y \diamond y : \phi$ by letting $Y = Z \diamond y$ and applying $\langle \phi? \rangle L$. By $(\langle * \rangle)$ on the right, it suffices to show $x \diamond Y : \langle \mathit{true}? \rangle \neg (\mathit{false}\, \mathbf{suf}\, \neg \phi)$, i.e., $x \diamond Z \diamond y \diamond y : \neg (\mathit{false}\, \mathbf{suf}\, \neg \phi)$. By $(\neg R)$:

$$Z \diamond y \diamond y : \phi, x \diamond Z \diamond y \diamond y : \mathit{false}\, \mathbf{suf}\, \neg \phi \Rightarrow \cdot.$$

By (**suf**), since $\mathit{false} \wedge (\cdot)$ is trivially false, we get $Z \diamond y \diamond y : \neg\phi$. Contradiction with $Z \diamond y \diamond y : \phi$. Closed.

For (xiii.a.*.2): By $(\langle * \rangle)$ on the right, it suffices to show $x \diamond Y : \langle \alpha_1 \rangle \langle \alpha_1^* \rangle \neg(\mathit{false}\,\mathbf{suf}\,\neg\phi)$.

By the induction hypothesis for α_1:

$$(\mathit{xiii.a.} * .21)\quad Y : \langle \alpha_1 \rangle \langle \alpha_1^* \rangle \phi \Rightarrow x \diamond Y : \langle \alpha_1 \rangle \neg(\mathit{false}\,\mathbf{suf}\,\neg\langle \alpha_1^* \rangle \phi)$$

is provable. We also need:

$$(\mathit{xiii.a.} * .22)\quad x \diamond Y : \langle \alpha_1 \rangle \neg(\mathit{false}\,\mathbf{suf}\,\neg\langle \alpha_1^* \rangle \phi) \Rightarrow x \diamond Y : \langle \alpha_1 \rangle \langle \alpha_1^* \rangle \neg(\mathit{false}\,\mathbf{suf}\,\neg\phi)$$

in order to prove (xiii.a.*.2) by (cut). By the Necessitation Lemma, it suffices to show:

$$x \diamond Y : \neg(\mathit{false}\,\mathbf{suf}\,\neg\langle \alpha_1^* \rangle \phi) \Rightarrow x \diamond Y : \langle \alpha_1^* \rangle \neg(\mathit{false}\,\mathbf{suf}\,\neg\phi)$$

By $(\neg R)$ and $(\neg L)$: $\cdot \Rightarrow x \diamond Y : \mathit{false}\,\mathbf{suf}\,\neg\langle \alpha_1^* \rangle \phi, x \diamond Y : \langle \alpha_1^* \rangle \neg(\mathit{false}\,\mathbf{suf}\,\neg\phi)$. By (**suf**) on $x \diamond Y : \mathit{false}\,\mathbf{suf}\,\neg\langle \alpha_1^* \rangle \phi$, since $\mathit{false} \wedge (\cdot)$ is trivially false, we need $Y : \neg\langle \alpha_1^* \rangle \phi$, i.e., by $(\neg R)$: $Y : \langle \alpha_1^* \rangle \phi \Rightarrow x \diamond Y : \langle \alpha_1^* \rangle \neg(\mathit{false}\,\mathbf{suf}\,\neg\phi)$, which is exactly (xiii.a.*), forming a back-link. The derivation path is progressive since the Necessitation Lemma has been applied.

Case for proving $(\mathit{xiii.b})$: That is, $X : (\mathit{false}\,\mathbf{suf}\,\mathit{true}) \wedge \langle \alpha \rangle \neg(\mathit{false}\,\mathbf{suf}\,\neg\phi) \Rightarrow X : \mathit{false}\,\mathbf{suf}\langle \alpha \rangle \phi$.

By $(\wedge L)$ on the left:

$$X : \mathit{false}\,\mathbf{suf}\,\mathit{true}, X : \langle \alpha \rangle \neg(\mathit{false}\,\mathbf{suf}\,\neg\phi) \Rightarrow X : \mathit{false}\,\mathbf{suf}\langle \alpha \rangle \phi$$

Without loss of generality, let $X = x \diamond Y$. By (**suf**) on $x \diamond Y : \mathit{false}\,\mathbf{suf}\,\mathit{true}$ on the left, since $\mathit{false} \wedge (\cdot)$ is trivially false, we get $Y : \mathit{true}$ (trivially satisfied). By (**suf**) on $x \diamond Y : \mathit{false}\,\mathbf{suf}\langle \alpha \rangle \phi$ on the right, since $\mathit{false} \wedge (\cdot)$ is trivially false, we need $Y : \langle \alpha \rangle \phi$. So it suffices to show:

$$x \diamond Y : \langle \alpha \rangle \neg(\mathit{false}\,\mathbf{suf}\,\neg\phi) \Rightarrow Y : \langle \alpha \rangle \phi$$

We prove this by structural induction on α.

Case $\alpha = a$ **(atomic action).** Let $Y = Z \diamond y$. We need:

$$x \diamond Z \diamond y : \langle a \rangle \neg(\mathit{false}\,\mathbf{suf}\,\neg\phi) \Rightarrow Z \diamond y : \langle a \rangle \phi$$

By $(\langle a \rangle L)$ on the left, we get $y \,\mathrm{R}_a\, z'$ and $x \diamond Z \diamond y \diamond z' : \neg(\mathit{false}\,\mathbf{suf}\,\neg\phi)$. By $(\langle a \rangle R)$ on the right (with the same z'), we need $Z \diamond y \diamond z' : \phi$. By $(\neg L)$ on $x \diamond Z \diamond y \diamond z' : \neg(\mathit{false}\,\mathbf{suf}\,\neg\phi)$, we need to show $\Rightarrow x \diamond Z \diamond y \diamond z' : \mathit{false}\,\mathbf{suf}\,\neg\phi, Z \diamond y \diamond z' : \phi$. By (**suf**) on $x \diamond Z \diamond y \diamond z' : \mathit{false}\,\mathbf{suf}\,\neg\phi$, since $\mathit{false} \wedge (\cdot)$ is trivially false, we need $Z \diamond y \diamond z' : \neg\phi$. So we need $\Rightarrow Z \diamond y \diamond z' : \neg\phi, Z \diamond y \diamond z' : \phi$, which closes by $(\neg R)$ and (ax).

Case $\alpha = \chi?$ **(test).** Let $Y = Z \diamond y$. We need:

$$x \diamond Z \diamond y : \langle \chi? \rangle \neg(\mathit{false}\,\mathbf{suf}\,\neg\phi) \Rightarrow Z \diamond y : \langle \chi? \rangle \phi$$

By $(\langle \phi? \rangle L)$ on the left, we get $y \diamond y : \chi$ and $x \diamond Z \diamond y \diamond y : \neg(\mathit{false}\,\mathbf{suf}\,\neg\phi)$. By $(\langle \phi? \rangle R)$ on the right, we need:

(xiii.b.χ?.1) $y \diamond y : \chi, x \diamond Z \diamond y \diamond y : \neg(\mathit{false}\,\mathbf{suf}\,\neg\phi) \Rightarrow y \diamond y : \chi$, closed by (ax).

(xiii.b.χ?.2) $y \diamond y : \chi, x \diamond Z \diamond y \diamond y : \neg(\mathit{false}\,\mathbf{suf}\,\neg\phi) \Rightarrow Z \diamond y \diamond y : \phi$. By $(\neg L)$ on $x \diamond Z \diamond y \diamond y : \neg(\mathit{false}\,\mathbf{suf}\,\neg\phi)$, then (**suf**), this reduces to $\Rightarrow Z \diamond y \diamond y : \neg\phi, Z \diamond y \diamond y : \phi$, which closes by $(\neg R)$ and (ax).

Case $\alpha = \alpha_1 ; \alpha_2$ **(sequential composition).** By $(\langle ; \rangle)$:

$$(\mathit{xiii.b.};.1)\quad x \diamond Y : \langle \alpha_1 \rangle \langle \alpha_2 \rangle \neg(\mathit{false}\,\mathbf{suf}\,\neg\phi) \Rightarrow Y : \langle \alpha_1 \rangle \langle \alpha_2 \rangle \phi$$

By induction hypothesis, we have that

$$(xiii.b.;.11)\ \ x \diamond Y : \langle\alpha_1\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\langle\alpha_2\rangle\phi) \Rightarrow Y : \langle\alpha_1\rangle\langle\alpha_2\rangle\phi$$

is provable. So by (cut), to prove (xiii.b.;.1), we still need

$$x \diamond Y : \langle\alpha_1\rangle\langle\alpha_2\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\phi) \Rightarrow x \diamond Y : \langle\alpha_1\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\langle\alpha_2\rangle\phi),$$

which is

$$(xiii.b.;.12)\ \ x \diamond Y : \langle\alpha_2\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\phi) \Rightarrow x \diamond Y : \neg(\mathit{false}\ \mathbf{suf}\ \neg\langle\alpha_2\rangle\phi)$$

by Necessitation Lemma. By (cut), (xiii.b.;.12) yields

(xiii.b.;.121) $x \diamond Y : \langle\alpha_2\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\phi) \Rightarrow Y : \langle\alpha_2\rangle\phi$, which is provable by induction hypothesis.

(xiii.b.;.122) $Y : \langle\alpha_2\rangle\phi \Rightarrow x \diamond Y : \neg(\mathit{false}\ \mathbf{suf}\ \neg\langle\alpha_2\rangle\phi)$, which can be proved by further extend $x \diamond Y : \neg(\mathit{false}\ \mathbf{suf}\ \neg\langle\alpha_2\rangle\phi)$ and obtain both $Y : \langle\alpha_2\rangle\phi$ and $Y : \neg\langle\alpha_2\rangle\phi$ on the left.

Case $\alpha = \alpha_1 \cup \alpha_2$ **(choice).** We need to prove

$$x \diamond Y : \langle\alpha_1 \cup \alpha_2\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\phi) \Rightarrow Y : \langle\alpha_1 \cup \alpha_2\rangle\phi.$$

By $(\langle\cup\rangle)$, $x \diamond Y : \langle\alpha_1 \cup \alpha_2\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\phi)$ gives $x \diamond Y : \langle\alpha_1\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\phi)$ or $x \diamond Y : \langle\alpha_2\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\phi)$. In the sub-case for α_1: by the induction hypothesis, we get $Y : \langle\alpha_1\rangle\phi$. Since $Y : \langle\alpha_1\rangle\phi$ implies $Y : \langle\alpha_1 \cup \alpha_2\rangle\phi$ (by $(\langle\cup\rangle)$), we can obtain $Y : \langle\alpha_1 \cup \alpha_2\rangle\phi$ by (cut). The sub-case for α_2 is symmetric.

Case $\alpha = \alpha_1^*$ **(iteration).** We need to prove

$$(viii.b.*)\ \ x \diamond Y : \langle\alpha_1^*\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\phi) \Rightarrow Y : \langle\alpha_1^*\rangle\phi.$$

By $(\langle*\rangle)$, $x \diamond Y : \langle\alpha_1^*\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\phi)$ gives $x \diamond Y : \langle\mathit{true}?\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\phi)$ or $x \diamond Y : \langle\alpha_1\rangle\langle\alpha_1^*\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\phi)$. We need $Y : \langle\alpha_1^*\rangle\phi$. We need to prove:

(viii.b.*.1) $x \diamond Y : \langle\mathit{true}?\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\phi) \Rightarrow Y : \langle\alpha_1^*\rangle\phi$

(viii.b.*.2) $x \diamond Y : \langle\alpha_1\rangle\langle\alpha_1^*\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\phi) \Rightarrow Y : \langle\alpha_1^*\rangle\phi$

Sub-case (viii.b.*.1): Let $Y = Z \diamond y$, by $(\langle\phi?\rangle L)$ we can replace $x \diamond Y : \langle\mathit{true}?\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\phi)$ with $y \diamond y : \mathit{true}$ and $x \diamond Z \diamond y \diamond y : \neg(\mathit{false}\ \mathbf{suf}\ \neg\phi)$. While on the left side, by $\langle*\rangle$, we obtain $Z \diamond y : \langle\mathit{true}?\rangle\phi$ and $Z \diamond y : \langle\alpha_1\rangle\langle\alpha_1^*\rangle\phi$. By $(\langle\phi?\rangle R)$ we can replace $Z \diamond y : \langle\mathit{true}?\rangle\phi$ with $Z \diamond y \diamond y : \phi$. The branch is then closed by extend $x \diamond Z \diamond y \diamond y : \neg(\mathit{false}\ \mathbf{suf}\ \neg\phi)$ and obtain $Z \diamond y \diamond y : \neg\phi$ on the right.

Sub-case (viii.b.*.2): By $(\langle*\rangle)$ on the right, it suffices to show $Y : \langle\alpha_1\rangle\langle\alpha_1^*\rangle\phi$. By (cut), to prove (viii.b.*.2), it is equivalent to prove:

(viii.b.*.21) $x \diamond Y : \langle\alpha_1\rangle\langle\alpha_1^*\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\phi) \Rightarrow x \diamond Y : \langle\alpha_1\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\langle\alpha_1^*\rangle\phi)$

(viii.b.*.22) $x \diamond Y : \langle\alpha_1\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\langle\alpha_1^*\rangle\phi) \Rightarrow Y : \langle\alpha_1\rangle\langle\alpha_1^*\rangle\phi$

note that $Y : \langle\alpha_1\rangle\langle\alpha_1^*\rangle\phi$ implies $Y : \langle\alpha_1^*\rangle\phi$ (by $(\langle*\rangle)$).

(viii.b.*.22) is provable by inductive hypothesis for α_1. From (viii.b.*.21), by Necessitation Lemma, we have

$$x \diamond Y : \langle\alpha_1^*\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\phi) \Rightarrow x \diamond Y : \neg(\mathit{false}\ \mathbf{suf}\ \neg\langle\alpha_1^*\rangle\phi)$$

By $(\neg R)$: $x \diamond Y : \langle\alpha_1^*\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\phi), x \diamond Y : \mathit{false}\ \mathbf{suf}\ \neg\langle\alpha_1^*\rangle\phi \Rightarrow \cdot$. By $(\mathbf{suf})$, since $\mathit{false} \wedge (\cdot)$ is trivially false, we get $Y : \neg\langle\alpha_1^*\rangle\phi$. By $(\neg L)$:

$$(viii.b.*.221)\ \ x \diamond Y : \langle\alpha_1^*\rangle\neg(\mathit{false}\ \mathbf{suf}\ \neg\phi) \Rightarrow Y : \langle\alpha_1^*\rangle\phi,$$

which is exactly (viii.b.*), forming a back-link. The derivation path from (viii.b.*) to (viii.b.*.221) is progressive since the Necessitation Lemma has been applied.

□

Proposition A.14 (Rule (xiv)). *Rule (xiv):* $\langle\alpha\rangle true \to \mathbf{fin}$ *is derived in G3PPL, where we define* $\mathbf{fin} =_{df} true\,\mathbf{suf}(true \wedge \neg\,\mathbf{n}\,true)$*, i.e.,*

$$\mathbf{fin} =_{df} true\,\mathbf{suf}(true \wedge \neg(false\,\mathbf{suf}\,true)),$$

which means there exists a finite suffix where the trace ends.

Proof. Let $X \in TVar$. Since X is finite, the premise $X : \langle\alpha\rangle true$ is in fact not used. By (wk) on the left, it suffices to prove the premise-free sequent

$$(xiv.1)\quad \cdot \Rightarrow X : true\,\mathbf{suf}(true \wedge \neg(false\,\mathbf{suf}\,true)).$$

We give a cyclic derivation of (xiv.1).

Without loss of generality, let $X = x \diamond Y$. By $(\mathbf{suf})$ on $x \diamond Y : true\,\mathbf{suf}(true \wedge \neg(false\,\mathbf{suf}\,true))$ on the right, we obtain:

$$\cdot \Rightarrow Y : (true \wedge \neg(false\,\mathbf{suf}\,true)) \vee (true \wedge true\,\mathbf{suf}(true \wedge \neg(false\,\mathbf{suf}\,true)))$$

By $(\vee R)$, this becomes:

$$\cdot \Rightarrow Y : (true \wedge \neg(false\,\mathbf{suf}\,true)), Y : true \wedge true\,\mathbf{suf}(true \wedge \neg(false\,\mathbf{suf}\,true))$$

By $(\wedge R)$ on $Y : true \wedge true\,\mathbf{suf}(true \wedge \neg(false\,\mathbf{suf}\,true))$ on the right, we split into two sub-cases:

(xiv.11) $\cdot \Rightarrow Y : (true \wedge \neg(false\,\mathbf{suf}\,true)), Y : true$, which is closed since we have $Y : true$ on the right.

(xiv.12) $\cdot \Rightarrow Y : (true \wedge \neg(false\,\mathbf{suf}\,true)), Y : true\,\mathbf{suf}(true \wedge \neg(false\,\mathbf{suf}\,true))$.

For (xiv.12), by (wk) we obtain:

$$(xiv.121)\quad \cdot \Rightarrow Y : true\,\mathbf{suf}(true \wedge \neg(false\,\mathbf{suf}\,true)),$$

which is exactly (xiv.12) with Y in place of X, forming a back-link. The derivation path from (xiv.1) to (xiv.121) is progressive since $(\mathbf{suf})$ has been applied.

□

Proposition A.15 (Rule (xv)). *Rule (xv):* $((\mathbf{n}\,\phi \to \phi)\,\mathbf{suf}\,\phi) \to \mathbf{n}\,\phi$ *is derived in G3PPL.*

Proof. Recall $\mathbf{n}\,\phi =_{df} false\,\mathbf{suf}\,\phi$. So we need to prove:

$$(xv.1)\quad X : ((false\,\mathbf{suf}\,\phi) \to \phi)\,\mathbf{suf}\,\phi \Rightarrow X : false\,\mathbf{suf}\,\phi.$$

Without loss of generality, let $X = x \diamond Y$. By $(\mathbf{suf})$ on $x \diamond Y : ((false\,\mathbf{suf}\,\phi) \to \phi)\,\mathbf{suf}\,\phi$ on the left:

$$Y : \phi \vee (((false\,\mathbf{suf}\,\phi) \to \phi) \wedge ((false\,\mathbf{suf}\,\phi) \to \phi)\,\mathbf{suf}\,\phi) \Rightarrow x \diamond Y : false\,\mathbf{suf}\,\phi$$

By $(\mathbf{suf})$ on $x \diamond Y : false\,\mathbf{suf}\,\phi$ on the right, since $false \wedge (\cdot)$ is trivially false, we need $Y : \phi$. By $(\vee L)$ on the left, we split into 2 cases:

(xv.11) $Y : \phi \Rightarrow Y : \phi$, which is closed by (ax).

(xv.12) $Y : ((false\,\mathbf{suf}\,\phi) \to \phi) \wedge ((false\,\mathbf{suf}\,\phi) \to \phi)\,\mathbf{suf}\,\phi \Rightarrow Y : \phi$

For case (xv.12), by $(\wedge L)$:

$$Y : (\mathit{false}\,\mathbf{suf}\,\phi) \to \phi, Y : ((\mathit{false}\,\mathbf{suf}\,\phi) \to \phi)\,\mathbf{suf}\,\phi \Rightarrow Y : \phi$$

By $(\to L)$ on $Y : (\mathit{false}\,\mathbf{suf}\,\phi) \to \phi$, we split into:

(xv.121) $Y : ((\mathit{false}\,\mathbf{suf}\,\phi) \to \phi)\,\mathbf{suf}\,\phi \Rightarrow Y : \mathit{false}\,\mathbf{suf}\,\phi, Y : \phi$.
By weakening on $Y : \phi$, this reduces to:

$$Y : ((\mathit{false}\,\mathbf{suf}\,\phi) \to \phi)\,\mathbf{suf}\,\phi \Rightarrow Y : \mathit{false}\,\mathbf{suf}\,\phi$$

which is exactly (xv.1) (with Y in place of X), forming a back-link. The derivation path is progressive since (**suf**) has been applied.

(xv.122) $Y : \phi, Y : ((\mathit{false}\,\mathbf{suf}\,\phi) \to \phi)\,\mathbf{suf}\,\phi \Rightarrow Y : \phi$, which is closed by (ax).

□

A.3 Completeness Proofs of G3FOPL

The proof of Lemma 7.2 requires two derived rules ($[*']$) and ($\langle *' \rangle$), whose unlabelled versions can be derived from the rules of FODL [12]. Below we show that they are derivable in G3FOPL.

Lemma A.1. *The following rules can be derived from G3FOPL:*

$$\frac{X : \Gamma \Rightarrow X : \varphi, X : \Delta \quad X : \varphi \Rightarrow X : [\alpha]\varphi \quad X : \varphi \Rightarrow X : \phi}{X : \Gamma \Rightarrow X : [\alpha^*]\phi, X : \Delta}\ ([*'])$$

$$\frac{X : \Gamma \Rightarrow X : \exists n \geq 0.\varphi(n), X : \Delta \quad X : \varphi(n+1) \Rightarrow X : \langle \alpha_1 \rangle \varphi(n) \quad X : \varphi(0) \Rightarrow X : \phi}{X : \Gamma \Rightarrow X : \langle \alpha^* \rangle \phi, X : \Delta}\ (\langle *' \rangle)$$

The proof of Lemma A.1 depends on Lemma A.2 below, which is itself established using Lemma 6.2 from Section 6. We therefore state and prove Lemma A.2 first.

Lemma A.2. *The following rules can be derived from G3FOPL:*

$$\frac{X : \phi \Rightarrow X : [\alpha]\phi}{X : \phi \Rightarrow X : [\alpha^*]\phi}\ (inv)$$

$$\frac{X : \phi(n+1) \Rightarrow X : \langle \alpha \rangle \phi(n)}{X : \phi(n) \Rightarrow X : \langle \alpha^* \rangle \phi(0)}\ (con)$$

In Lemma A.2, rule (inv) corresponds to the *loop invariance rule* of [12], while rule (con) corresponds to the *convergence rule* of [12].

Proof of Lemma A.2. The derivations for rules (inv) and (con) are given below; both rely on rule $([*])$ to unfold the iteration and on the Necessitation Lemma to handle modalities.

The derivation for rule (inv):

$$\dfrac{\dfrac{\text{Lemma 6.2}}{X:\phi \Rightarrow X:[\mathit{true}?]\phi} \quad \dfrac{X:\phi \Rightarrow X:[\alpha]\phi \quad \dfrac{X:\phi \Rightarrow X:[\alpha^*]\phi \text{ (bud)}}{X:[\alpha]\phi \Rightarrow X:[\alpha][\alpha^*]\phi}\ (nec)}{X:\phi \Rightarrow X:[\alpha][\alpha^*]\phi}\ (cut)}{\dfrac{X:\phi \Rightarrow X:[\mathit{true}? \cup \alpha;\alpha^*]\phi}{X:\phi \Rightarrow X:[\alpha^*]\phi}\ ([*])}\ ([\cup])$$

The derivation for rule (con):

$$\dfrac{\dfrac{\text{Lemma 6.2}}{X:n=0, X:\phi(0) \Rightarrow X:\langle \mathit{true}?\rangle\phi(0)} \quad \dfrac{\text{Cont.}}{X:n\geq 1, X:\phi(n) \Rightarrow X:\langle\alpha\rangle\langle\alpha^*\rangle\phi(0)}}{\dfrac{\dfrac{X:\phi(n) \Rightarrow X:\langle\alpha\rangle\langle\alpha^*\rangle\phi(0), X:\langle \mathit{true}?\rangle\phi(0)}{X:\phi(n) \Rightarrow X:\langle \mathit{true}? \cup \alpha;\alpha^*\rangle\phi(0)}\ ([*])}{X:\phi(n) \Rightarrow X:\langle\alpha^*\rangle\phi(0)}\ [\cup]},$$

$$\dfrac{X:\phi(n+1) \Rightarrow X:\langle\alpha\rangle\phi(n) \quad \dfrac{X:\phi(n) \Rightarrow X:\langle\alpha^*\rangle\phi(0) \text{ (bud)}}{X:\langle\alpha\rangle\phi(n) \Rightarrow X:\langle\alpha\rangle\langle\alpha^*\rangle\phi(0)}\ (nec)}{\text{Cont.}: X:\phi(n+1) \Rightarrow X:\langle\alpha\rangle\langle\alpha^*\rangle\phi(0)}\ (cut),$$

where note that $X : n \geq 1 \wedge \phi(n)$ is logically equivalent to $X : \phi(n+1)$ (where we assume $n \geq 0$); from $X : \phi(0) \Rightarrow X : \langle \mathit{true}?\rangle\phi(0)$ one can also make use of Lemma 6.2 by firstly applying rule $(\neg R)$ and $(\neg L)$ to transform the sequent into the required form.

□

We now give the derivations of the two rules stated in Lemma A.1.

Proof of Lemma A.1. **Derivation of rule $([*'])$ in Lemma A.1.** Starting from

$$(*'.a)\ \ X:\Gamma \Rightarrow X:[\alpha^*]\phi, X:\Delta,$$

by applying (cut), we need:

$(*'.\text{a}.1)$ $X:\Gamma \Rightarrow X:\varphi, X:[\alpha^*]\phi, X:\Delta$, from which by (wk) we obtain

$$(*'.a.11)\ \ X:\Gamma \Rightarrow X:\varphi, X:\Delta$$

$(*'.\text{a}.2)$ $X:\Gamma, X:\varphi \Rightarrow X:[\alpha^*]\phi, X:\Delta$.

From $(*'.\text{a}.2)$, by (cut) and (wk), we need:

$(*'.\text{a}.21)$ $X:\varphi \Rightarrow X:[\alpha^*]\varphi$, from which by rule (inv) in Lemma A.2, we have

$$(*'.a.211)\ \ X:\varphi \Rightarrow X:[\alpha]\varphi.$$

$(*'.\text{a}.22)$ $X:[\alpha^*]\varphi \Rightarrow X:[\alpha^*]\phi$, from which by Necessitation Lemma (Lemma 6.1), we have

$$(*'.a.221)\ \ X:\varphi \Rightarrow X:\phi$$

We have obtained the premisses of the rule $([*'])$ as: $(*'.\text{a}.11)$, $(*'.\text{a}.211)$, $(*'.\text{a}.221)$.

Derivation of rule $(\langle *'\rangle)$ in Lemma A.1. Starting from

$$(*'.b)\ \ X:\Gamma \Rightarrow X:\langle\alpha^*\rangle\phi, X:\Delta,$$

By applying (cut), we need:

$(*'.b.1)$ $X : \Gamma \Rightarrow X : \exists n.\varphi(n) \wedge n \geq 0, X : \langle\alpha^*\rangle\phi, X : \Delta$, from which by (wk) we obtain

$$(*'.b.11)\ \ X : \Gamma \Rightarrow X : \exists n.\varphi(n) \wedge n \geq 0, X : \Delta$$

$(*'.b.2)$ $X : \Gamma, X : \exists n.\varphi(n) \wedge n \geq 0 \Rightarrow X : \langle\alpha^*\rangle\phi, X : \Delta$

From $(*'.b.2)$, by $(\exists L)$, $(\wedge L)$ and (wk), we have

$$(*'.b.21)\ \ X : \varphi(n) \Rightarrow X : \langle\alpha^*\rangle\phi.$$

By (cut), we need:

$(*'.b.211)$ $X : \varphi(n) \Rightarrow X : \langle\alpha^*\rangle\varphi(0)$, from which by rule (con) in Lemma A.2, we obtain

$$(*'.b.2111)\ \ X : \varphi(n+1) \Rightarrow X : \langle\alpha\rangle\varphi(n)$$

$(*'.b.212)$ $X : \langle\alpha^*\rangle\varphi(0) \Rightarrow X : \langle\alpha^*\rangle\phi$, from which, by Necessitation Lemma (Lemma 6.1), we need

$$(*'.b.2121)\ \ X : \varphi(0) \Rightarrow X : \phi$$

We have obtained the premisses of the rule $(\langle*'\rangle)$ as: $(*'.b.11)$, $(*'.b.2111)$, $(*'.b.2121)$. □

Proof of Lemma 7.2. We proceed by induction on the structure of the program α. We first consider the case when op is $[\alpha]$, the case when op is $\langle\alpha\rangle$ is similar except for the case of star programs.

The base case is when α is an atomic program, namely a. Without loss of generality, let $X = Y \diamond x$. To prove $Y \diamond x : \phi^\flat \Rightarrow Y \diamond x : [a]\psi^\flat$, by rule $([\alpha]R)$, it is sufficient to prove

$$Y \diamond x : \phi^\flat, xR_a y \Rightarrow Y \diamond x \diamond y : \psi^\flat \tag{5}$$

By $\models (\phi^\flat \rightarrow [a]\psi^\flat)$, it is not hard to see that (5) is valid. So by (ter), we close the derivation.

The case when α is a sequential program, namely $\alpha_1 ; \alpha_2$. By the soundness of rule $([;])$, we obtain $\models X : \phi^\flat \Rightarrow X : [\alpha_1][\alpha_2]\psi^\flat$. By Lemma 7.1, there exists a pure arithmetical FOL formula $\varphi^\flat$ such that $\models [\alpha_2]\psi^\flat \leftrightarrow \varphi^\flat$. Hence $\models X : \varphi^\flat \Rightarrow X : [\alpha_2]\psi^\flat$ and $\models X : \phi^\flat \Rightarrow X : [\alpha_1]\varphi^\flat$. By induction hypothesis, we have $\vdash X : \varphi^\flat \Rightarrow X : [\alpha_2]\psi^\flat$ and $\vdash X : \phi^\flat \Rightarrow X : [\alpha_1]\varphi^\flat$. By Necessitation Lemma, since $\vdash X : \varphi^\flat \Rightarrow X : [\alpha_2]\psi^\flat$, $\vdash X : [\alpha_1]\varphi^\flat \Rightarrow X : [\alpha_1][\alpha_2]\psi^\flat$. By it and $\vdash X : \phi^\flat \Rightarrow X : [\alpha_1]\varphi^\flat$, using rule (cut), we obtain $\vdash X : \phi^\flat \Rightarrow X : [\alpha_1][\alpha_2]\psi^\flat$. The result is direct by rule $([;])$.

The case when α is a choice program, namely $\alpha_1 \cup \alpha_2$. By the soundness of rule $([\cup])$, $\models X : \phi^\flat \Rightarrow X : ([\alpha_1]\psi^\flat \wedge [\alpha_2]\psi^\flat)$, which is equivalent to $\models X : \phi^\flat \Rightarrow X : [\alpha_1]\psi^\flat$ and $\models X : \phi^\flat \Rightarrow X : [\alpha_2]\psi^\flat$. By induction hypothesis, we have $\vdash X : \phi^\flat \Rightarrow X : [\alpha_1]\psi^\flat$ and $\vdash X : \phi^\flat \Rightarrow X : [\alpha_2]\psi^\flat$. Therefore $\vdash X : \phi^\flat \Rightarrow X : [\alpha_1 \cup \alpha_2]\psi^\flat$ by rule $([\cup])$.

The case when α is a star program, namely α_1^*. By Lemma 7.1, there exists an arithmetical FOL formula $\varphi^\flat$ such that $\models [\alpha_1^*]\psi^\flat \leftrightarrow \varphi^\flat$. Then from $\models X : \phi^\flat \Rightarrow X : [\alpha_1^*]\psi^\flat$, we also have $\models X : \phi^\flat \Rightarrow X : \varphi^\flat$. From $\models [\alpha_1^*]\psi^\flat \leftrightarrow \varphi^\flat$, by the soundness of the rules $([*])$, $([\cup])$, $([\phi?])$ and $([;])$, it is not hard to see that $\models \varphi^\flat \leftrightarrow [\alpha_1^*]\psi^\flat \leftrightarrow [true? \cup \alpha_1 ; \alpha_1^*]\psi^\flat \leftrightarrow \psi^\flat \wedge [\alpha_1][\alpha_1^*]\psi^\flat \leftrightarrow \psi^\flat \wedge [\alpha_1]\varphi^\flat$. From these logical equivalences we see that $\models \varphi^\flat \rightarrow [\alpha_1]\varphi^\flat$ and $\models \varphi^\flat \rightarrow \psi^\flat$. By induction hypothesis, from $\models X : \phi^\flat \Rightarrow X : \varphi^\flat$, $\models X : \varphi^\flat \Rightarrow X : [\alpha_1]\varphi^\flat$, and $\models X : \varphi^\flat \Rightarrow X : \psi^\flat$, we have $\vdash X : \phi^\flat \Rightarrow X : \varphi^\flat$, $\vdash X : \varphi^\flat \Rightarrow X : [\alpha_1]\varphi^\flat$ and $\vdash X : \varphi^\flat \Rightarrow X : \psi^\flat$. By rule $([*'])$, we have $\vdash X : \phi^\flat \Rightarrow X : [\alpha_1^*]\psi^\flat$.

Now we consider the case when op is $\langle\alpha\rangle$, where the situations for α is an atomic, sequential and choice program are similar to those for the case when op is $[\alpha]$ by applying the corresponding rules of the rules used above. The only non-trivial case is when α is a star program α_1^*.

By Lemma 7.1 and the way of expressing regular programs in arithmetical FOL in [12], we know that for any $n \geq 0$, $\langle \alpha_1^n \rangle \psi^\flat$ can be expressed as a formula $\varphi^\flat(n)$, i.e., $\models \langle \alpha_1^n \rangle \psi^\flat \leftrightarrow \varphi^\flat(n)$. From the semantics of $\langle \alpha_1^* \rangle \psi^\flat$, it is easy to see that $\models \langle \alpha_1^* \rangle \psi^\flat \leftrightarrow \exists n \geq 0.\varphi^\flat(n)$. From $\models X : \phi^\flat \Rightarrow X : \langle \alpha_1^* \rangle \psi^\flat$, we have $\models X : \phi^\flat \Rightarrow X : \exists n \geq 0.\varphi^\flat(n)$. On the other hand, by the soundness of the rule $(\langle ; \rangle)$, there is $\varphi^\flat(n+1) \leftrightarrow \langle \alpha_1^{n+1} \rangle \psi^\flat \leftrightarrow \langle \alpha_1 ; \alpha_1^n \rangle \psi^\flat \leftrightarrow \langle \alpha_1 \rangle \langle \alpha_1^n \rangle \psi^\flat \leftrightarrow \langle \alpha_1 \rangle \varphi^\flat(n)$. From these logical equivalences we get that $\models \varphi^\flat(n+1) \rightarrow \langle \alpha_1 \rangle \varphi^\flat(n)$. When $n = 0$, from $\models \langle \alpha_1^n \rangle \psi^\flat \leftrightarrow \varphi^\flat(n)$, by the soundness of rule $(\langle \phi ? \rangle)$, we have $\models \langle q^0 \rangle \psi^\flat \leftrightarrow \langle \mathit{true} ? \rangle \psi^\flat \leftrightarrow \psi^\flat \leftrightarrow \varphi^\flat(0)$. So $\models \varphi^\flat(0) \rightarrow \psi^\flat$. By induction hypothesis, from $\models X : \phi^\flat \Rightarrow X : \exists n \geq 0.\varphi^\flat(n)$, $\models X : \varphi^\flat(n+1) \Rightarrow X : \langle \alpha_1 \rangle \varphi^\flat(n)$ and $\models X : \varphi^\flat(0) \Rightarrow X : \psi^\flat$, we have $\vdash X : \phi^\flat \Rightarrow X : \exists n \geq 0.\varphi^\flat(n)$, $\vdash X : \varphi^\flat(n+1) \Rightarrow X : \langle \alpha_1 \rangle \varphi^\flat(n)$ and $\vdash X : \varphi^\flat(0) \Rightarrow X : \psi^\flat$. By rule $\langle * ' \rangle$, we obtain $\vdash X : \phi^\flat \Rightarrow X : \langle \alpha_1^* \rangle \psi^\flat$. □