Appears in a special issue of the *Journal of Applied Non-Classical Logics* on Interval Temporal Logics and Duration Calculi (volume 14, no. 1–2, pages 55–104). Guest editors: Valentin Goranko and Angelo Montanari.

A Hierarchical Completeness Proof for Propositional Interval Temporal Logic with Finite Time

Ben Moszkowski

Software Technology Research Laboratory
Hawthorn Building
De Montfort University
The Gateway
Leicester LE1 9BH (Great Britain)

benm@dmu.ac.uk

ABSTRACT. *We present a completeness proof for* Propositional Interval Temporal Logic (PITL) *with finite time which avoids certain difficulties of conventional methods. It is more gradated than previous efforts since we progressively reduce reasoning within the original logic to simpler reasoning in sublogics. Furthermore, our approach benefits from being less constructive since it is able to invoke certain theorems about regular languages over finite words without the need to explicitly describe the associated intricate proofs.*

A modified version of regular expressions called Fusion Expressions *is used as part of an intermediate logic called* Fusion Logic. *Both have the same expressiveness as* PITL *but are lower-level notations which play an important role in the hierarchical structure of the overall completeness proof. In particular, showing completeness for* PITL *is reduced to showing completeness for Fusion Logic. This in turn is shown to hold relative to completeness for conventional linear-time temporal logic with finite time.*

Logics based on regular languages over finite words and ω-words offer a promising but elusive framework for formal specification and verification. A number of such logics and decision procedures have been proposed. In addition, various researchers have obtained complete axiom systems by embedding and expressing the decision procedures directly within the logics. The work described here contributes to this topic by showing how to exploit some interesting links between regular languages and interval-based temporal logics.

KEYWORDS: *interval temporal logic, completeness proof, fusion product.*

1. Introduction

Interval Temporal Logic (ITL) [MOS 83b, HAL 83, MOS 85] is a temporal logic with a basic construct for the sequential composition of two formulas as well as an analogue of Kleene star. Within ITL, one can express both finite-state automata and regular expressions. Its notation makes it suitable for logic-based modular reasoning involving periods of time, refinement [CAU 97], sequential composition using assumptions and commitments based on fixpoints of temporal operators [MOS 94, MOS 98] and for executable specifications [MOS 86]. Various imperative programming constructs are expressible in ITL and projection between time granularities is available (but not considered here). Zhou, Hoare and Ravn [Zho 91] present an ITL extension called *Duration Calculus* for hybrid systems (see also Zhou and Hansen [Zho 04]). Several researchers have looked at ITL decision procedures and axiom systems.

We present a natural and complete axiomatisation for (quantifier-free) Propositional ITL (PITL) for finite time. The completeness proof is hierarchical and shows that if a PITL formula A is logically consistent (i.e., not provably false), then there exists a certain consistent formula X in conventional Propositional Temporal Logic (PTL). Now the completeness of PTL ensures that this X has a satisfying model. This model in turn also satisfies the original PITL formula A. Our proof exploits a variant of regular expressions called here *fusion expressions*. The completeness proof also uses an associated subset of PITL called here *Fusion Logic* (FL) which acts as a bridge between PITL and PTL.

In previous work on complete axiom systems for ITL with finite time [MOS 00a] and infinite time [MOS 00b], we made use of embeddings of finite-state automata within ITL formulas. Consequently, we had to use quantified PITL and include axioms for it in order to reason about hidden automata states. This approach was motivated by an earlier automata-based completeness proof of Kesten and Pnueli [KES 95, KES 02] for Quantified PTL (QPTL) with past time. However, in the course of preparing the full versions of our papers, we observed that regular expressions (and subsequently fusion expressions) offer some important advantages in PITL completeness proofs over automata. Among other things, they are closer to the ITL notion and, at least with finite time, totally avoid the need for quantified variables which can hide automata states. More importantly, they reduce the need for representing within formulas a number of constructions involving automata. However, in the case of infinite time, automata seem unavoidable. We note that French and Reynolds [FRE 03] have recently proved the completeness of an axiom system for QPTL without past time.

A recent and related paper of ours [MOS 04a] uses intervals and consistency-based reasoning in a new hierarchical and compositional proof of completeness for PTL. Taken together, the papers provide evidence of the strong symbiotic links between PTL and PITL. Some of the techniques described in these papers have been have been further developed in another work by us [MOS 04b] which gives a proof of the small model property for *Propositional Dynamic Logic* (PDL) [FIS 77, FIS 79, HAR 84, KOZ 90, HAR 00, HAR 02] without using the conventional Fischer-Ladner

closures introduced in [FIS 77, FIS 79]. This supports our belief that research on intervals and ITL, while challenging, is worthwhile. It seems to even offer unexpected interesting spin-offs which increase our understanding of the theory of other logics.

1.1. *Structure of Presentation*

The organisation of this work is now summarised. Section 2 reviews related work. Section 3 gives an overview of the syntax and semantics of PITL. Section 4 presents the PITL axiom system. Section 5 deals with some useful notions of completeness and relative completeness which are utilised in our hierarchical method. Section 6 describes regular languages and fusion languages. Both are required for analysing Fusion Logic (FL) introduced later in this work. Section 7 defines fusion expressions (also called FE formulas) and looks at their expressiveness. Section 8 describes FL which acts as an intermediate logic between PITL and PTL. Section 9 introduces an axiom system for FL which is proved to be complete in a later section. Section 10 presents the PTL axiom system which later serves as a basis for successively showing completeness for the FL and PITL axiom systems. Section 11 considers FE formula behaviour in certain useful classes of intervals. This is necessary for subsequently reducing FL formulas containing the iterative construct chop-star which is akin to Kleene-star. Section 12 describes a way to indirectly characterise chop-star in FL. This later plays a key role in obtaining relative completeness of useful FL subsets. Section 13 presents a proof of completeness for the FL axiom system by transforming formulas into suitable PTL ones and then making use of the completeness of the PTL axiom system. Section 14 deals with embedding the FL and PTL axiom systems in the PITL axiom system. Section 15 establishes the completeness of the PITL axiom system. We exploit the expressiveness of fusion expressions to reduce PITL completeness to FL completeness. Section 16 concludes with some discussion. Appendix A presents some results concerning the complexity of PITL. Appendix B shows how to deduce PTL axioms and inference rules in the FL axiom system.

1.2. *Diagrammatic Representation of Parts of Completeness Proof*

Let us now present a rough summary of the interrelationship of some key parts of the completeness proof in diagrammatic form. A reader might wish to refer back to this as he studies this work. The notation $S \rightarrow S'$ found in the diagrams means that the set of formulas S is a subset of the set of formulas S' and furthermore completeness for the elements of S is shown by us to imply completeness for the elements of S'. If an arrow has a number on its top, then the number refers to the relevant theorem or lemma which explicitly establishes the indicated result or is at least a very significant link in proving it.

We commence with the axiomatic completeness of PTL (Theorem 51) but rely on a proof of it published elsewhere. Therefore the first major stage actually proved here deals with going from PTL completeness to FL completeness. A countably infinite

sequence of subsets of FL formulas denoted here as $\mathrm{FL}_V^{0,0}$, $\mathrm{FL}_V^{0,1}$, ... are used. They are later formally introduced in Definition 44.

$$\mathrm{PTL} \xrightarrow{67} \mathrm{FL}_V^{0,0} \longrightarrow \cdots \longrightarrow \mathrm{FL}_V^{n,m} \xrightarrow{68} \mathrm{FL}_V^{n,m+1}$$
$$\longrightarrow \cdots \xrightarrow{69} \mathrm{FL}_V^{n+1,0} \longrightarrow \cdots \xrightarrow{70} \mathrm{FL}_V \xrightarrow{71} \mathrm{FL} \ .$$

Below is the structure of the proof from FL to PITL using countably infinite subsets of PITL formulas denoted as PITL^0, PITL^1, ... and formally introduced later in Definition 80:

$$\mathrm{FL} \xrightarrow{84a} \mathrm{PITL}^0 \cdots \longrightarrow \cdots \mathrm{PITL}^n \xrightarrow{84b} \mathrm{PITL}^{n+1} \longrightarrow \cdots \xrightarrow{85} \mathrm{PITL} \ .$$

Here Lemma 84 has a proof by induction. The suffix "a" refers to the base case and the suffix "b" refers to the inductive proof step.

2. Related Work

Let us now discuss other work on axiom systems for ITL and then some closely related calculi. A proof of completeness for such notations is often based on some kind of decision procedure so we make some mention of this as well. Halpern and Moszkowski [MOS 83a, pages 23–24] prove the decidability of PITL with quantifiers over finite time by translation to QPTL over finite time which is decidable by an easy modification of an analogous result for conventional QPTL over infinite time by Wolper [WOL 82, SIS 87] (see also [LIC 85] for a direct proof). The satisfiability problem for PITL has nonelementary complexity and hence is much harder than that for popular logics such as PTL. We include statements and proofs of relevant results for PITL in Appendix A. These difficulties with complexity have also manifested themselves in work on complete axiom systems for PITL. The topic seems to present more hurdles than in the case for other some related logics. The reader should bear this in mind when attempting to assess progress in this area.

Rosner and Pnueli [ROS 86] investigate an axiom system for quantifier-free PITL with finite and ω-intervals and the *until* operator. However it does not contain the operator *chop-star* which is like Kleene-star for regular expressions. A tableau method serves as the decision procedure underlying the completeness proof and employs an adaptation of Fischer-Ladner closures developed for Propositional Dynamic Logic (PDL) [FIS 79, HAR 00]. One of the inference rules is quite large and requires constructing an *index-table* containing *indices* (including *terminal indices*) and an *accessibility relation* for automata transitions connected with tableau construction. Furthermore, the inference rule necessitates deducing three categories of PITL theorems concerning accessibility between indices in the index-table before an inference can actually be made.

Paech [PAE 88] investigates a quantifier-free version of PITL with ω-intervals having *chop-star* limited, like Kleene-star, to a finitely many iterations and including an additional temporal operator *unless*. Due to a theorem of Thomas [THO 79]

(later more simply proved by Y. Choueka and D. Peleg [CHO 83]), PITL with such a restricted *chop-star* is still as expressive as ω-regular expressions (and hence the logics S1S [BÜC 62] and QPTL) as well as quantifier-free propositional PITL with unrestricted *chop-star* (which permits ω consecutive finite iterations) although with possibly less succinctness. An additional temporal operator *unless*, which is a variant of the conventional operator *until*, is also included. Paech presents a complete Gentzen-style proof system which includes some nonconventional axioms which obligate certain PITL formulas to already be in a form analogous to regular expressions. This can potentially involve complex meta-reasoning about arbitrary PITL formulas over finite intervals to ensure suitability for these particular axioms. The proof method is adapted from one used by Nishimura [NIS 79] for PDL and subsequently refined by Valiev [VAL 79]. Consequently, a generalised form of Fischer-Ladner closures is necessary to cope with negation and other aspects of PITL not found in PDL programs. Surprisingly, the axioms, unlike those of Rosner and Pnueli, appear to limit intervals to be infinite. Therefore no modular reasoning about finite subintervals is possible.

Dutertre [DUT 95] gives two complete proof systems for first-order ITL without *chop-star* for finite time. The first uses a possible-worlds semantics of time and the second considers arbitrary linear orderings of states. Neither is complete for standard discrete-time intervals. Wang Hanpin and Xu Qiwen [Wan 99] generalise Dutertre's results to handle infinite time. Kono [KON 95] presents a tableau-based decision procedure for PITL with quantifiers and temporal projection over finite time which has been successfully implemented in Prolog[1]. No formal proof is given that the method does not overlook models. Instead, a sketchy argument about termination is presented. Kono suggests that the transformations provide a partial basis for a complete axiom system. Many details are omitted and one of the two proposed axioms for projection is unsound[2]. Moszkowski [MOS 94] presents propositional and first-order axiom systems for ITL over finite intervals. This is shown to support proofs involving sequential and parallel aspects of compositionality based on the *rely-guarantee* paradigm of Jones [JON 83]. The propositional part is claimed to be complete but only a brief outline of a proof is given. This work is extended in [MOS 95] to include a axiomatisation for temporal projection which is complete relative to PITL without projection.

Bowman and Thompson [BOW 98] present a detailed study of a tableau-based decision procedure for quantifier-free PITL with finite time and temporal projection but do not give an axiom system. They omit considerations about the termination of their method. In [BOW 00, BOW 03] they look at termination and also obtain a completeness proof for an axiomatisation of this version of PITL.

Wolper [WOL 83] presents *Extended Temporal Logic* (ETL) which includes operators containing explicit automata descriptions. This make ETL's expressive power equivalent to that of S1S. A decision procedure and complete axiom system are given. Compared with PITL, ETL's notational reliance on automata makes it less suited for

1. We have extensively used it and never found a bug.
2. However, Kono has told us that the associated problem in the implemented decision procedure was rectified early on.

sequential composition. Banieqbal and Barringer [BAN 86] show the unsoundness of one ETL inference rule and offer a transition-based remedy. Henriksen and Thiagarajan [HEN 97, HEN 99] investigate a formalism related to ETL and called *Dynamic Linear Time Temporal Logic* which combines PTL and PDL in a linear-time framework with infinite time. It is similar to our Fusion Logic and uses multiple atomic programs instead of tests containing propositional formulas. The axiom system has special inference rules involving transitions and sets of words.

So far we have reviewed various temporal logics. Let us now look at certain related formalisms. Some mention has already be given of Siefkes' pioneering work [SIE 70] on proving a axiom system for S1S to be complete. He achieved the proof by embedding a decision procedure developed by Büchi [BÜC 62] in S1S itself through a process he called *syntactization*. This even included a challenging proof of a restricted but nontrivial version of Ramsey's theorem.

Salomaa [SAL 66] examines axiom systems for equational theories of regular expressions over finite words. Wagner [WAG 76] extends Salomaa's work to handle an axiom system for equations between ω-regular expressions without an operator for complementing. The system is proved in detail to be complete through the use of ω-automata. Kozen [KOZ 94] presents an complete finitary axiomatisation of *Kleene algebras* involving equations and inferences between equations. The completeness proof utilises an algebraic encoding of classical operators on finite-state automata such as determinisation and state minimisation. Regular expressions over finite words provide the motivation for Kleene algebras and serve as the standard models for them.

Many problems in computing can be modelled by means of words over a finite nonempty alphabet. Finite-state machines and regular expressions are perhaps the best known ways of describing such words. These normally only handle finite words but Büchi's seminal work [BÜC 62] showed how to extend both concepts to handle ω-words by means of *Büchi-automata* and ω-regular expressions. As seen above, various researchers have investigated logics which are tailored for reasoning ω-words. Lichtenstein, Pnueli and Zuck [LIC 85] as well as Emerson [EME 90] survey the relationship between a number of such formalisms. Decision procedures, complete axioms systems, various results about expressiveness and succinctness as well as some software tools exist. However, unlike regular expressions and PITL, the logics so far mentioned do not have basic facilities for reasoning about arbitrary subwords and lack a simple way to sequentially compose two or more formulas about such subwords to obtain a formula for an overall word. Nor is some analogue of Kleene-star generally available. For example, conventional temporal logics often provide a construct for examining the next state and another construct for examining all states before some particular condition is true. However, one can not take two arbitrary temporal formulas and combine them one after the other as is routinely done with the concatenation of regular expressions. Although the binary temporal-logic operator known as *until* offers some degree of sequential composition, the left operand must be specially expressed to achieve this. In addition, unlike concatenation, *until* lacks associativity.

3. Overview of Propositional Interval Temporal Logic

We now briefly describe (quantifier-free) propositional ITL (PITL) for finite time. More details about ITL are in [MOS 83b, MOS 83a, HAL 83, MOS 85, MOS 86, MOS 94, MOS 98, MOS 00a]. Time in PITL is discrete and linear. An interval σ has an *interval length* $|\sigma| \geq 0$ and a finite, nonempty sequence of $|\sigma| + 1$ states $\sigma_0, \ldots, \sigma_{|\sigma|}$. There is a denumerable set Var of propositional variables such as P, Q and R. Each state σ_i maps a variable in Var such as P to a value $\sigma_i(P)$. In PITL the only values are $true$ and $false$. The set of all finite-time PITL intervals will be denoted here as INT.

Here is the syntax of PITL's formulas, where A and B are themselves formulas:

$$true \quad v \text{ (for propositional variable } v) \quad \neg A \quad A \vee B \quad skip \quad A; B \quad A^*.$$

There are three primitive temporal operators:

$$skip \qquad A; B \ \textit{(chop)} \qquad A^* \ \textit{(chop-star)} \ ,$$

where A and B are themselves formulas. The formula $skip$ is true on a two-state interval. A formula $A; B$ is true on σ iff σ can be chopped into two subintervals sharing a state σ_k for some $k \leq |\sigma|$ with A true on $\sigma_0 \ldots \sigma_k$ and B true on $\sigma_k \ldots \sigma_{|\sigma|}$. Thus the formula $skip; P$ is true on σ iff σ has at least two states and P is true in σ_1. A formula A^* is true on σ iff σ can be chopped into zero or more parts with A true on each. Any formula A^* (including $false^*$) is true on a one-state interval (see Subsect. 3.1). Figure 1 pictorially illustrates the semantics of *skip*, *chop*, and *chop-star*. Some simple PITL formulas together with intervals which satisfy them are shown in Fig. 2. In the figure, the logical values $true$ and $false$ are respectively abbreviated as "`t`" and "`f`". Furthermore, for some sample formulas we include in parentheses versions using conventional temporal logic operators which are formally introduced later in Subsect. 3.3.

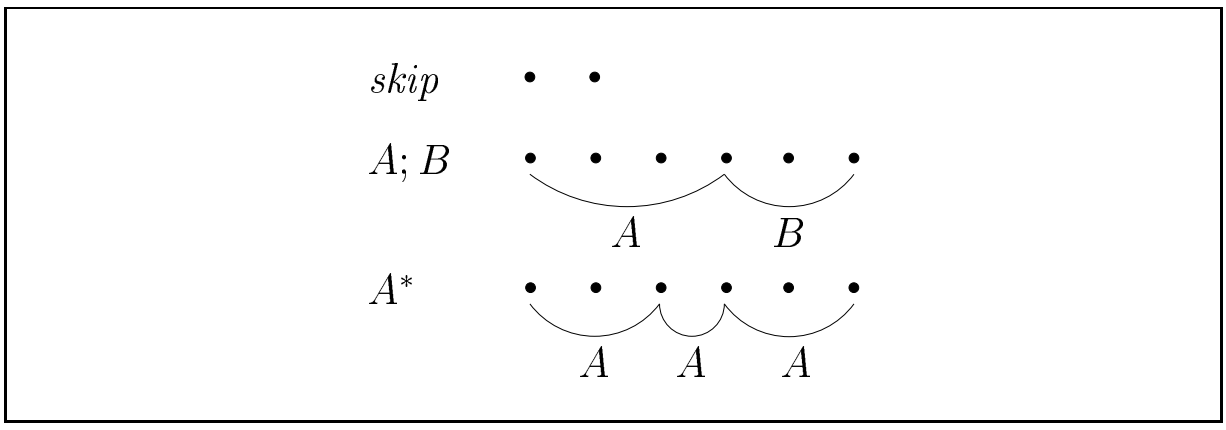

Figure 1. *Informal semantics*

For natural numbers i, j with $i \leq j \leq |\sigma|$, let $\sigma_{i:j}$ denotes the subinterval having interval length $j - i$ (i.e., $j - i + 1$ states) and with starting state σ_i and final state σ_j.

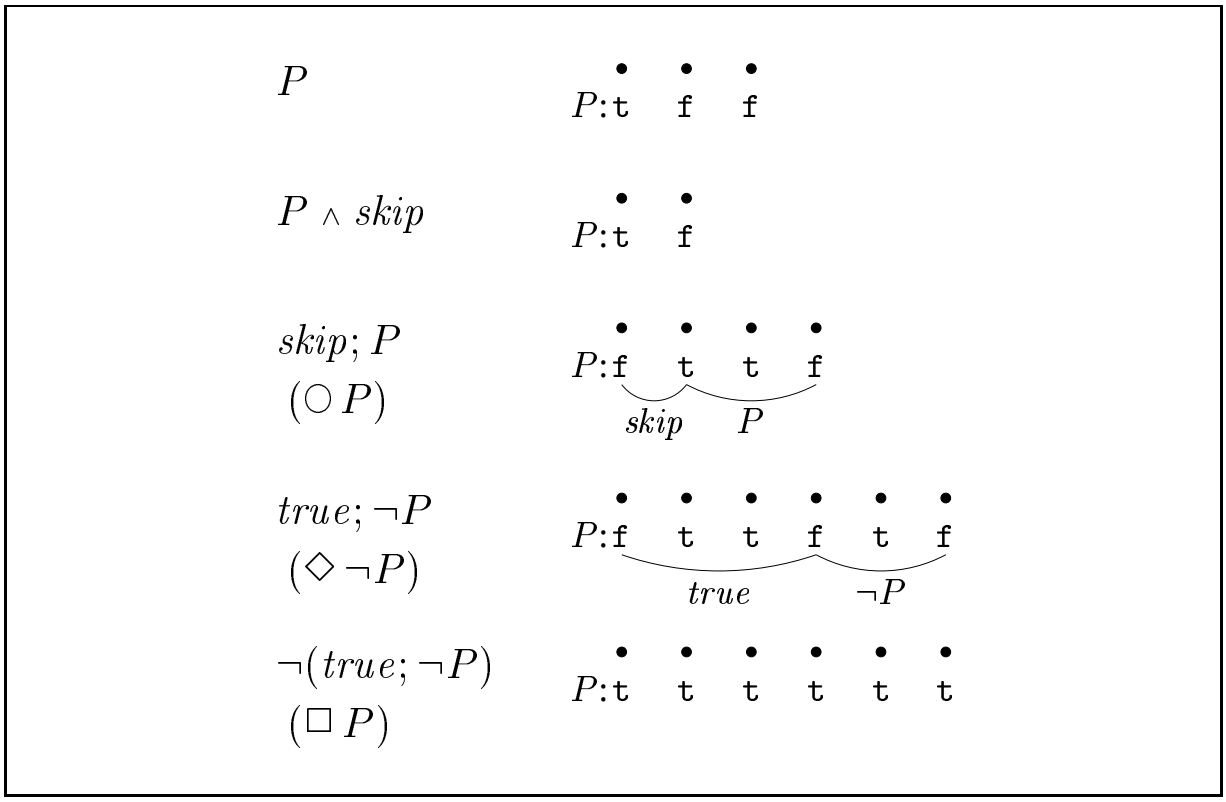

Figure 2. *Some sample* PITL *formulas and intervals*

Below is the syntax and semantics of the PITL constructs used here. We denote the semantics of a formula A on interval σ as $\mathcal{M}_\sigma[\![A]\!]$.

3.1. *Semantics of Formulas*

– $\mathcal{M}_\sigma[\![true]\!] = true$.

– $\mathcal{M}_\sigma[\![P]\!] = \sigma_0(P)$, for any propositional variable $P \in Var$.
The value of P for an interval σ is P's value in σ's initial state σ_0.

– $\mathcal{M}_\sigma[\![\neg A]\!] = true$ iff $\mathcal{M}_\sigma[\![A]\!] = false$.

– $\mathcal{M}_\sigma[\![A \vee B]\!] = true$ iff $\mathcal{M}_\sigma[\![A]\!] = true$ or $\mathcal{M}_\sigma[\![B]\!] = true$.

– $\mathcal{M}_\sigma[\![skip]\!] = true$ iff $|\sigma| = 1$.

– $\mathcal{M}_\sigma[\![A; B]\!] = true$ iff $\mathcal{M}_{\sigma'}[\![A]\!] = true$ and $\mathcal{M}_{\sigma''}[\![B]\!] = true$,
where $\sigma' = \sigma_{0:i}$ and $\sigma'' = \sigma_{i:|\sigma|}$ for some $i \leq |\sigma|$. Intervals σ' and σ'' share state σ_i.

– $\mathcal{M}_\sigma[\![A^*]\!] = true$ iff $\mathcal{M}_{\sigma_{l_i:l_{i+1}}}[\![A]\!] = true$, for each $i : 0 \leq i < n$,
for some $n \geq 0$ and finite sequence of natural numbers $l_0 \leq l_1 \leq \cdots \leq l_n$ where $l_0 = 0$ and $l_n = |\sigma|$.

DEFINITION 1 (EMPTY INTERVALS). — *We refer to intervals containing exactly one state as* empty intervals.

The behaviour of chop-star on empty intervals is a frequent source of confusion. We therefore state the following important lemma which is simple to prove:

LEMMA 2 (CHOP-STAR AND EMPTY INTERVALS). — *For any* PITL *formula* A, *every empty interval satisfies* A^*.

PROOF. — In the definition of chop-star's semantics we can let $n = 0$ no matter what A is. ■

Consequently, even $false^*$ is true in any empty interval, whereas it is false in all other intervals. The PDL program $(false?)^*$ exhibits similar behaviour.

DEFINITION 3 (SATISFIABILITY AND VALIDITY). — *If* $\mathcal{M}_\sigma[\![A]\!] = true$ *for* some *interval* σ, *then* σ satisfies A, *denoted* $\sigma \models A$. *In addition,* A *is said to be* satisfiable.

A formula A *satisfied by all intervals is* valid, *denoted* $\models A$.

Notation such as $\models_{\mathrm{PITL}} A$ can used to avoid ambiguity when properties of different logics are being compared.

3.2. *Conventions Regarding Variable Names*

Variables A, B, C and variants such as A' denote PITL formulas. The variable w denotes a *state formula* containing no temporal operators.

Later on, we use variables X, Y, Z and variants such as X' to denote formulas in certain subsets of PITL such as conventional Propositional Temporal Logic (PTL).

3.3. *Some Definable Constructs*

Boolean constructs such as $false$ and $A \wedge B$ are definable as are the conventional temporal constructs $\Diamond A$ ("sometimes A"), $\Box A$ ("always A") and $\bigcirc A$ ("next A"):

$$\Diamond A \stackrel{\text{def}}{\equiv} true; A \qquad \Box A \stackrel{\text{def}}{\equiv} \neg \Diamond \neg A \qquad \bigcirc A \stackrel{\text{def}}{\equiv} skip; A \ .$$

The sublogic built only from the temporal operators $\Diamond$ and $\bigcirc$ is commonly known as *Propositional Temporal Logic (*PTL*).* Here are more operators expressible in this:

$\circledcirc A$	$\stackrel{\text{def}}{\equiv} \neg \bigcirc \neg A$	(Weak next)	$fin\ A$	$\stackrel{\text{def}}{\equiv} \Box(empty \supset A)$	(Final state)
$more$	$\stackrel{\text{def}}{\equiv} \bigcirc true$	(More states)	$finite$	$\stackrel{\text{def}}{\equiv} \Diamond empty$	(Finite time)
$empty$	$\stackrel{\text{def}}{\equiv} \neg more$	(One state)	$\boxdot A$	$\stackrel{\text{def}}{\equiv} \Box(more \supset A)$	(Mostly)

The conventional temporal operator *until* is definable if quantification is added but it is not needed here. A restricted version can be expressed as $(\boxdot A); B$, where the the only temporal constructs in the subformula A are non-nested $\bigcirc$-operators. This suffices for many purposes. Alternatively, *until* can be readily added as a primitive operator. Below are operators for examining *initial* and *arbitrary* subintervals:

$$\diamondsuit A \stackrel{\text{def}}{\equiv} A; true \qquad \boxminus A \stackrel{\text{def}}{\equiv} \neg \diamondsuit \neg A \qquad \circledcirc A \stackrel{\text{def}}{\equiv} true; A; true \qquad \boxtimes A \stackrel{\text{def}}{\equiv} \neg \circledcirc \neg A \ .$$

Appendix A presents some results on PITL's complexity. Decidability is nonelementary but is elementary in the alternation of ¬ and chop (and chop-star).

4. PITL Axiom System

We now present an axiom system for PITL. Our experience in rigorously developing hundreds of proofs has helped us refine the axioms and convinced us of their utility for a wide range of purposes such as compositional reasoning [MOS 94, MOS 98].

DEFINITION 4 (TAUTOLOGY). — *A tautology is any formula which is a substitution instance of some valid nonmodal propositional formula.*

For example, any PITL formula of the form $\bigcirc A \vee \Diamond B \supset \Diamond B$ is a tautology since it is a substitution instance of the valid nonmodal formula $P \vee Q \supset Q$. It is not hard to show that all tautologies are themselves valid. Intuitively, a formula is a tautology if it does not require any modal reasoning to justify its truth.

4.1. *Axioms and Inference Rules for* PITL

Our PITL axiom system is given in Table 1 and first appeared in [MOS 94]. Recall that the symbol $\supset$ is the logical operator *implication* used in formulas. In contrast, the metalogical symbol $\Rightarrow$ denotes the ability to infer a new theorem from other previously deduced ones. The axiom system mainly deals with *chop*, and *skip* and operators derived from them. Only one axiom is needed for *chop-star* (but see Remarks 6).

Table 1. *Axiom system for* PITL

Taut	$\vdash$ All PITL tautologies	**P7**	$\vdash\ w \supset \boxdot w$
P2	$\vdash\ (A;B);C \equiv A;(B;C)$	**P8**	$\vdash\ \boxdot(A \supset A') \wedge \Box(B \supset B')$ $\supset\ (A;B) \supset (A';B')$
P3	$\vdash\ (A \vee A');B\ \supset\ (A;B) \vee (A';B)$		
P4	$\vdash\ A;(B \vee B')\ \supset\ (A;B) \vee (A;B')$	**P9**	$\vdash\ \bigcirc A \supset \circledw A$
P5	$\vdash\ empty;A\ \equiv\ A$	**P10**	$\vdash\ A \wedge \Box(A \supset \circledw A)\ \supset\ \Box A$
P6	$\vdash\ A;empty\ \equiv\ A$	**P11**	$\vdash\ A^*\ \equiv$ $empty\ \vee\ (A \wedge more);A^*$
MP	$\vdash\ A \supset B,\ \ \vdash\ A\ \ \Rightarrow\ \ \vdash\ B$		
□Gen	$\vdash\ A\ \ \Rightarrow\ \ \vdash\ \Box A$	**⊡Gen**	$\vdash\ A\ \ \Rightarrow\ \ \vdash\ \boxdot A$

The axiom system contains some of the propositional axioms suggested by Rosner and Pnueli [ROS 86] but also includes our own axioms and inference rule for the operators ⊡ and *chop-star*. These assist in deducing theorems and derived inference rules for compositional reasoning (see our work in [MOS 94, MOS 98] for more details). The Axiom **Taut** permits using properties of conventional nonmodal logic without proof (recall Definition 4 concerning tautologies). It is possible to omit it and achieve

the same results by means of a few "lower-level" axioms and inference rules dealing primarily with nonmodal reasoning.

The axiom system gives nearly equal treatment to initial and terminal subintervals. For example, the Inference Rules $\boxdot$**Gen** and $\Box$**Gen** respectively provide a means to obtain new theorems by embedding previously deduced PITL theorems in $\boxdot$ and $\Box$. This is exceedingly important for the kinds of proofs we do since we naturally move formulas in and out of the left side of chop in many situations. The later embedding of the FL axiom system in the PITL axiom system and the reduction of PITL completeness to FL completeness both involve a lot of this kind of reasoning. The proof of the PITL Replacement Theorem (Theorem 77) is also a good example of how the analysis of the left side of chop is relevant. We additionally believe that axioms and inference rules concerning $\boxdot$ make the axiom system easier to understand since much of it consists simply of duals in this sense. In contrast, most temporal logics cannot readily handle initial subintervals since the conventional operators are point-based. Even other axiom systems for ITL largely neglect initial subintervals.

A formula A which is deducible (provable) from the axioms and inferences rules is called an PITL *theorem*, denoted $\vdash_{\mathrm{PITL}} A$ or simply $\vdash A$. When doing proofs, we can observe that a PITL subset in which the only primitive temporal operator is chop and one side is always some fixed formula obeys the rules of the conventional normal modal system K (see Chellas [CHE 80] and Hughes and Cresswell [HUG 96]). We now give two sample theorems and their proofs. The justification **Prop** in some steps refers to conventional propositional reasoning which can involve implicit uses of Axiom **Taut** and/or modus ponens.

T1 $\vdash \quad \boxdot(A \supset B) \quad \supset \quad \diamondsuit A \supset \diamondsuit B$

Proof:

1	$\vdash$	$true \quad \supset \quad true$	**Prop**
2	$\vdash$	$\Box(true \supset true)$	1,$\Box$**Gen**
3	$\vdash$	$\boxdot(A \supset B) \wedge \Box(true \supset true)$ $\supset \quad (A; true) \supset (B; true)$	**P8**
4	$\vdash$	$\boxdot(A \supset B) \quad \supset \quad (A; true) \supset (B; true)$	2,3,**Prop**
5	$\vdash$	$\boxdot(A \supset B) \quad \supset \quad \diamondsuit A \supset \diamondsuit B$	4,def. of $\diamondsuit$

T2 $\vdash \quad finite$

Proof:

1	$\vdash$	$true; empty \quad \equiv \quad true$	**P6**
2	$\vdash$	$true; empty$	1,**Prop**
3	$\vdash$	$\diamond\, empty$	2,def. of $\diamond$
4	$\vdash$	$finite$	3,def. of $finite$

The following instance of Axiom **P7** illustrates why it is not subsumed by Inference Rule $\boxdot$**Gen**:

$$\vdash \quad \neg Q \supset \boxdot \neg Q \ .$$

Here Q is a propositional variable. We cannot use $\boxdot$**Gen** since $\neg Q$ is not a theorem.

LEMMA 5 (SOUNDNESS OF THE PITL AXIOM SYSTEM). — *The* PITL *axiom system is sound, that is, any formula which is a theorem is also valid.*

PROOF. — This involves induction on the length of a PITL theorem's finite proof in the axiom system. The base case involves guaranteeing that every concrete instance of each axiom is valid. This can be checked from the PITL semantics. The inductive case requires ensuring that if the inference rules are supplied theorems which are inductively assumed to be valid, then the resulting theorems are also valid. For example, consider Inference Rule $\boxdot$**Gen** which from $\vdash A$ yields $\vdash \boxdot A$. If the PITL formula A is a theorem, then by induction it is valid and true for all finite intervals. Therefore, it is true in all subintervals of any finite interval and hence $\boxdot A$ is true for all finite intervals and consequently itself valid. ■

REMARKS 6. — If desired, PITL Axiom **P11** concerning chop-star can be replaced by the following two axioms which some readers may find more natural:

$$\vdash_{\mathrm{PITL}} \ A^* \ \equiv \ \mathit{empty} \ \vee \ (A; A^*) \qquad \vdash_{\mathrm{PITL}} \ A^* \ \equiv \ (A \wedge \mathit{more})^* \ .$$

The second axiom can be optionally weakened to be just $A^* \supset (A \wedge \mathit{more})^*$. We omit the relevant proofs. However, readers interested in this should consult the material in Subsect. 14.3 concerning some properties of chop-star in PITL.

5. Notions of Completeness

Various notions connected with arbitrary logics and their axiom systems are now described. These notions facilitate showing deductive completeness in a modular way. Consequently, this section does not specifically concern PITL.

DEFINITION 7 (COMPLETENESS). — *A logic is* complete *if each valid formula is deducible as a theorem in the logic's axiom system. In other words, if* $\models A$*, then* $\vdash A$*.*

DEFINITION 8 (CONSISTENCY). — *We define a formula* A *to be* consistent *if* $\neg A$ *is* not *a theorem, i.e.,* $\nvdash \neg A$*.*

We will make use of the following standard variant way of expressing completeness by means of consistency:

LEMMA 9 (ALTERNATIVE NOTION OF COMPLETENESS). — *A logic's axiom system is complete iff each consistent formula is satisfiable.*

In the course of proving completeness we make use of definitions of completeness and relative completeness for sets of formulas:

DEFINITION 10 (COMPLETENESS FOR A SET OF FORMULAS). — *An axiom system is said to be complete for a set of formulas* $\{A_1, \ldots, A_n\}$ *if the consistency of any* A_i *implies its satisfiability.*

DEFINITION 11 (RELATIVE COMPLETENESS FOR A SET OF FORMULAS). — *The axiom system is said to be complete for one set of formulas* $\{A_1, \ldots, A_m\}$ *relative to another set of formulas* $\{B_1, \ldots, B_n\}$ *if completeness for the second set implies completeness for the first set.*

If a set has only one formula we will refer to completeness for the formula itself.

LEMMA 12 (TRANSITIVITY OF RELATIVE COMPLETENESS). — *Let* S_1, S_2 *and* S_3 *be three sets of formulas for which completeness holds for* S_1 *relative to formula* S_2 *and additionally for* S_2 *relative to* S_3. *Then completeness holds for* S_1 *relative to* S_3.

PROOF. — Suppose completeness holds for S_3. Then by relative completeness for S_2, it also holds for S_2. Subsequently, it holds for S_1 as well. Hence completeness holds for S_1 relative to S_3. ■

LEMMA 13. — *Suppose* A *and* B *are two formulas with the following properties:*

(a) If A *is consistent, then so is* B.

(b) If B *is satisfiable, then so is* A.

(c) Completeness holds for B.

Then completeness holds for A.

PROOF. — Suppose the assumptions are true. We need to show that if A is consistent then it is also satisfiable. Now if A is consistent, then by assumption (a), B is also consistent. In addition, assumption (c) now ensures that B is satisfiable as well. This together with assumption (b) guarantees A's satisfiability. Hence completeness holds for A. ■

COROLLARY 14 (COMPLETENESS FOR A FORMULA RELATIVE TO ANOTHER). — *Suppose* A *and* B *are two formulas with the following properties:*

– If A *is consistent, then so is* B.

– If B *is satisfiable, then so is* A.

Then completeness holds for A *relative to* B.

PROOF. — The proof is immediate from Lemma 13 and our definition of relative completeness. ■

DEFINITION 15 (DEDUCTIVE EQUIVALENCE OF TWO FORMULAS). — *Two formulas* A *and* B *are said to be* deductively equivalent *if the equivalence* $A \equiv B$ *is itself a deducible theorem (i.e.,* $\vdash A \equiv B$*).*

LEMMA 16 (RELATIVE COMPLETENESS AND DEDUCTIVE EQUIVALENCE). — *If two formulas* A *and* B *are deductively equivalent, then completeness holds for* A *relative to* B.

PROOF. — We use Corollary 14 and need to show that if A is consistent, then so is B and also if B is satisfiable, then so is A. The contrapositive of each will be proved.

Case for consistency: Suppose B is not consistent. Then $\neg B$ is a theorem. From the assumption $\vdash A \equiv B$ and simple propositional reasoning we readily establish that $\neg A$ is a theorem and hence A is not consistent.

Case for satisfiability: Suppose A is not satisfiable. Then $\neg A$ is valid. From the assumption $\vdash A \equiv B$ and the soundness of the axiom system we have $\models A \equiv B$ and hence $\models \neg B$. Therefore B is also not satisfiable. ■

6. Regular Languages and Fusion Languages

There is a natural connection between conventional regular languages and PITL since PITL formulas can express exactly those regular languages not containing the empty word. We use this in the proof of deductive completeness and require the notion of regular languages. Hamaguchi et al. [HIR 92, HAM 92] proposed versions of temporal logic with regular expressions. More recently, the language Sugar [BEE 01] has been developed at IBM's Haifa Research Laboratory for specifying functional properties of logic designs. It supports temporal logic with regular expressions and serves as the basis of an IEEE international standard. We will describe *fusion languages* which are similar to regular languages but are better suited for a version of temporal logic with close links to PITL. At least from a technical standpoint, fusion languages generally subsume regular languages. The formalism Dynamic Linear Time Temporal Logic of Henriksen and Thiagarajan [HEN 97, HEN 99] combines PTL and PDL in a linear-time framework with infinite time. It is similar to our formalism for fusion languages and uses multiple atomic programs instead of tests containing propositional formulas.

6.1. *An Alphabet Based on Propositional Variables*

A conventional regular language has a finite alphabet with no presumed internal structure. In contrast to this, we impose some natural restrictions due to our underlying logical framework. Let V be a finite set of zero or more propositional variables. Assuming V contains exactly n elements, we obtain an alphabet Σ_V whose letters are each of the 2^n possible subsets of V. In other words, Σ_V is the power set 2^V. For example, the set of variables $\{P, Q\}$ has the following four-element alphabet:

$$\{\} \qquad \{P\} \qquad \{Q\} \qquad \{P, Q\} \ .$$

We let λ, λ', etc. denote letters in Σ_V.

A *word* is now defined to be a sequence of zero or more letters in Σ_V. We only consider words having a finite number of letters. Define Σ_V^* to denote the set of all such words and Σ_V^+ to denote the set of all words having at least one letter. Now let

ε denote the (unique) empty word having zero letters and let each letter λ in Σ_V also denote the associated one-letter word. Given an alphabet Σ_V, a *language* over Σ_V is a set of words, that is, a subset of Σ_V^*. We readily obtain several trivial but important languages, namely, the empty language of no words denoted $\emptyset$, the singleton language $\{\varepsilon\}$ containing the empty word, and for each letter λ, the singleton language $\{\lambda\}$.

6.2. *Regular Languages*

For any two languages L and L', one can define the standard operators $L \cup L'$ (union), $L \cdot L'$ (concatenation) and L^* (Kleene star). We will often write $L \cdot L'$ more concisely as LL'. Here $L \cup L'$ denotes the set union of all words in L and L'. The language $L \cdot L'$ equals the set of all words obtained by pairwise concatenation of any word in L with any word in L'. Finally, L^* equals $\bigcup_{k \geq 0} L^k$, where L^0 is defined to the set containing just the empty word (i.e., $L^0 = \{\varepsilon\}$) and for any $n \geq 0$, L^{n+1} is defined to be $L \cdot L^n$.

DEFINITION 17 (THE SET Reg_V OF REGULAR LANGUAGES WITH ALPHABET Σ_V). — *We now define the set* Reg_V *of regular languages over finite words with alphabet* Σ_V *as follows (see Hopcroft and Ullman [HOP 79]):*

– *The empty language* $\emptyset$ *and the singleton language* $\{\varepsilon\}$ *are both in* Reg_V.

– *For each* λ *in* Σ_V*, the singleton language* $\{\lambda\}$ *is in* Reg_V.

– *If* L *and* L' *are in* Reg_V*, so are* $L \cup L'$*,* LL' *and* L^*.

– *These are the only languages in* Reg_V.

The following is a well known result of language theory:

LEMMA 18 (CLOSURE OF Reg_V UNDER COMPLEMENTATION). — *The complement of any language* L *in* Reg_V *with respect to* Σ_V^* *(i.e.,* $\Sigma_V^* \backslash L$*) is also in* Reg_V.

DEFINITION 19 (THE SET Reg_V^+). — *We define* Reg_V^+ *to be the set of languages in* Reg_V *not containing the empty word.*

LEMMA 20 (CLOSURE OF Reg_V^+ UNDER COMPLEMENTATION). — *The complement of any language* L *in* Reg_V^+ *with respect to* Σ_V^+ *(i.e.,* $\Sigma_V^+ \backslash L$*) is also in* Reg_V^+.

6.3. *Fusion Languages*

Before formally defining fusion languages, we define the *fusion* of two words:

DEFINITION 21 (FUSION PRODUCT OF TWO WORDS). — *Let* z *and* z' *be two nonempty words for which the last letter of* z *equals the first letter of* z'*. We let the* fusion product *of* z *and* z'*, denoted here as* $z \circ z'$*, be the word obtained by appending the two words together so as to include only one copy of the shared letter.*

For example, if z is the two-letter word $\lambda\lambda'$ and z' is the two-letter word $\lambda'\lambda''$, then their fusion $z \circ z'$ equals the three-letter word $\lambda\lambda'\lambda''$. If z and z' are compatible for fusion, then the number of letters in $z \circ z'$ is the sum of the number of letters in z and z' minus 1. Pratt first defined fusion product for describing the semantics of a process logic [PRA 79]. In what follows, we simply refer to this operation as *fusion*.

DEFINITION 22 (FUSION-COMPATIBLE WORDS). — *If z and z' are two nonempty words and the last letter of z equals the first letter of z', then we say that they are* fusion-compatible.

DEFINITION 23 (FUSION OF TWO LANGUAGES). — *Let L and L' be two languages. Their fusion, $L \circ L'$ is defined to be the language containing exactly all possible fusions involving fusion-compatible elements of L and L':*

$$L \circ L' \quad \stackrel{\text{def}}{=} \quad \{z \circ z' : z \in L \text{ and } z' \in L' \text{ and } z \text{ and } z' \text{ are fusion-compatible}\} \ .$$

We also need to define a version of Kleene star adapted to fusion. For any language L, let $L^{[0]}$ denote the language Σ_V containing exactly all words with one letter. The actual value of L is irrelevant in determining $L^{[0]}$. Further, for any $n \geq 0$, let $L^{[n+1]}$ denote the language fusion $L \circ L^{[n]}$.

DEFINITION 24 (FUSION STAR – A FUSION VERSION OF KLEENE STAR). — *For any language L, the fusion version of Kleene star applied to L is denoted as $L^{[*]}$. It is called here* fusion star *and is defined to be the union of all the languages $L^{[0]}$, $L^{[1]}$, ...:*

$$L^{[*]} \quad \stackrel{\text{def}}{=} \quad \bigcup_{i \geq 0} L^{[i]} \ .$$

DEFINITION 25 (THE SET Fusion_V OF FUSION LANGUAGES WITH ALPHABET Σ_V). — *We now define the set Fusion_V of fusion languages over finite words with alphabet Σ_V as follows:*

– *The empty language $\emptyset$ is in Fusion_V.*
– *For each λ in Σ_V, the singleton language $\{\lambda\}$ is in Fusion_V.*
– *The language $\Sigma_V\Sigma_V$, also denoted as Σ_V^2, of all two-letter words is in Fusion_V.*
– *If L and L' are in Fusion_V, so are $L \cup L'$, $L \circ L'$ and $L^{[*]}$.*
– *These are the only languages in Fusion_V.*

The next lemma compares regular and fusion languages in more detail:

LEMMA 26 (RELATIONSHIP BETWEEN REGULAR AND FUSION LANGUAGES). — *For any language $L \subseteq \Sigma_V^*$, the following are equivalent:*

(a) *L is in Reg_V^+.*
(b) *L is in Fusion_V.*

PROOF. — Finite-state automata assist in carrying out the proof although we do not need to explicitly describe them here in temporal logic formulas.

(a)⇒(b): We can construct a finite-state automaton which recognises L and then obtain a fusion language which equals the set of (nonempty) words accepted by the automaton.

(b)⇒(a): We can construct a finite-state automaton which recognises L and then obtain from it a regular language in Reg_V^+ which captures the (nonempty) words which the automaton accepts. ■

REMARKS 27. — A convenient variant of finite state automata specifically for fusion languages can be defined by slightly modifying the acceptance condition. We do not deal with this here but details can be found in our earlier automata-based proof of completeness for finite-time PITL [MOS 00a].

LEMMA 28 (CLOSURE OF FUSION LANGUAGES UNDER COMPLEMENTATION). — *The complement of any language L in Fusion_V with respect to Σ_V^+ (i.e., $\Sigma_V^+ \backslash L$) is also in Fusion_V.*

PROOF. — Lemma 26 ensures that L is also an element of Reg_V^+. Let L' be the complement of L with respect to Σ_V^+. By Lemma 20, L' is also in Reg_V^+. By another application of Lemma 26 to L', we reach our goal of showing that L' is in Fusion_V. ■

6.4. PITL *and Languages*

We now look at the connection between words, languages, intervals and PITL formulas.

DEFINITION 29 (THE SET OF FORMULAS PITL_V). — *For any set of variables V, let PITL_V denote the set of* PITL *formulas containing only variables in V.*

DEFINITION 30 (WORDS FROM INTERVALS AND LANGUAGES FROM PITL FORMULAS). — *For a given interval $\sigma \in \mathrm{INT}$, let $\sigma|_V$ denote the unique word in Σ_V^+ corresponding to the behaviour of V's variables in all of σ's states.*

Let the function $\mathcal{L}_V^+ : \mathrm{PITL}_V \to \Sigma_V^+$ map each formula A in PITL_V to an associated language which is a subset of Σ_V^+ as given below:

$$\mathcal{L}_V^+(A) \quad \stackrel{\mathrm{def}}{=} \quad \{\sigma|_V : \sigma \in \mathrm{INT} \text{ and } \sigma \models A\}.$$

DEFINITION 31 (DEFINABILITY OF A LANGUAGE IN A SET OF FORMULAS). — *Suppose S is a set of formulas. A language $L \subseteq \Sigma_V^+$ is said to be* definable *in S exactly if there exists some formula A in S for which L equals $\mathcal{L}_V^+(A)$.*

LEMMA 32 (EXPRESSIVENESS COMPLETENESS OF PITL_V). — *For any language $L \subseteq \Sigma_V^+$, the following are equivalent:*

(a) L *is in* Fusion_V.

(b) L *is definable in* PITL_V.

PROOF. — *(a)⇒(b)*: Each case in Definition 25 of fusion languages can be inductively expressed as a formula in PITL_V.

(b)⇒(a): Let A be a formula in PITL_V which expresses L. We do induction on A's syntax. The only nontrivial case is when A's main operator is negation so A has the form $\neg B$ for some PITL_V formula B. Lemma 28 ensures that if B defines a language in Fusion_V, then the complement of this language with respect to Σ_V^+ is also in Fusion_V. This ensures that $\mathcal{L}_V^+(A)$ is in Fusion_V. ■

7. Fusion Expressions

Regular expressions are a standard notation for representing regular languages. In our completeness proof, it is more appropriate to use fusion languages and a variation of regular expressions called here *fusion expressions*. We now define a PITL-based representation of them which is in fact a special subset of PITL formulas and plays a major role in our completeness proof.

DEFINITION 33 (FUSION EXPRESSION FORMULAS). — *The set of* fusion expression formulas*, denoted* FE*, consists of* PITL *formulas with the syntax given below, where* E *and* F *themselves denote such* FE *formulas:*

$$w? \qquad E \vee F \qquad \mathit{skip} \qquad E;F \qquad E^* \quad .$$

The new construct $w?$ *is defined below in* PITL*:*

$$w? \quad \stackrel{\mathrm{def}}{=} \quad w \wedge \mathit{empty} \quad .$$

The syntax of FE *formulas is like that of programs in Propositional Dynamic Logic (*PDL*) [FIS 79,KOZ 90,HAR 00] with a single atomic program* skip *and without rich tests. However* FE *has a semantics based on sequences of states rather than binary relations.*

For any set of variables V*, let* FE_V *denote the set of* FE *formulas containing only variables in* V.

Unlike letters in conventional regular expressions, any nonmodal formula can be used in $w?$. For example, $\mathit{false}?$ is permitted even though it is unsatisfiable. In contrast to $w?$, the other FE constructs are already found in PITL. Consider for example the following FE formula:

$$\big(\mathit{skip};(P \wedge Q)?\big) \vee \big((\neg Q)?;\mathit{skip}\big)^* .$$

This is true on an interval if either the interval has exactly two states and P and Q are both true in the second state or it has some arbitrary number of states, say k, with Q false in each of the first $k-1$ states.

The elements of FE_V in a certain sense express exactly all languages in Fusion_V. The following lemma formalises this:

LEMMA 34 (CORRESPONDENCE BETWEEN LANGUAGES IN Fusion_V AND FE_V FORMULAS). — *For any language L, the following are equivalent:*

(a) *L is in* Fusion_V.

(b) *L is definable in* FE_V.

PROOF. — We first prove the implication (a)⇒(b). Now each of the primitive fusion languages has a direct analogue in FE_V. Induction on the (finite) depth of operations in L's construction yields a corresponding element E in FE_V.

Now consider the implication (b)⇒(a). The goal here is to show that for each E in FE_V, the associated language $\mathcal{L}_V^+(E)$ is indeed in Fusion_V. The proof is by easy induction on E's syntax. ■

REMARKS 35. — It is worth pointing out that, as in PDL, programming language constructs such as conditional statements and while-loops can be expressed as fusion expressions. For example, $\mathit{while}\ w\ \mathit{do}\ E$ can be expressed as $(w?;E)^*;\neg w?$.

8. Fusion Logic

We now introduce a sublogic of PITL called here *Fusion Logic* (FL) which plays a central role in this paper. In essence, FL augments conventional PTL with fusion expressions. The establishment of deductive completeness of the PITL axiom system later on in Section 15 in Theorem 86 is first reduced in Lemma 85 to the task of showing deductive completeness of PITL relative to another axiom system limited to the sublogic FL. Prior to this, Theorem 71 deals with completeness for FL.

DEFINITION 36 (FUSION LOGIC). — *Here is the syntax of* FL *where P is any propositional variable in Var, E is any* FE *formula and X and Y are themselves formulas in* FL*:*

$$P \qquad \neg X \qquad X \vee Y \qquad \bigcirc X \qquad \Diamond X \qquad \langle E\rangle X.$$

*We define the new construct $\langle E\rangle X$ (called "*FL-chop*") and its dual $[E]X$ (called "*FL-yields*") by means of the primitive* PITL *constructs chop and $\neg$:*

$$\langle E\rangle X \stackrel{\mathrm{def}}{\equiv} E;X \qquad [E]X \stackrel{\mathrm{def}}{\equiv} \neg\langle E\rangle\neg X.$$

Within an FL formula, $\bigcirc$, $\Diamond$ and FL-chop will themselves be treated as primitive constructs. Other PTL operators such as empty and fin are expressible in FL in terms of $\bigcirc$ and $\Diamond$ (see Subsect. 3.3).

In contrast to PITL, FL limits the left sides of chop to being fusion expressions. Its syntax is like that of formulas in PDL. However FL has a semantics based on

sequences of states rather than binary relations. We have not done an analysis of FL's computational complexity.

DEFINITION 37 (LEFT-FORMULAS). — *For any element X of* FL*, we say that an* FE *formula E is a* left-formula *of X if E occurs as the lefthand operand of one of X's* FL*-chop constructs. In other words, E is a left-formula in X iff for some* FL *formula Y, X is itself the* FL *formula $\langle E \rangle Y$ or contains this as a subformula.*

For example, let X be the formula shown below:

$$(\langle P?; Q? \rangle \langle Q?^* \vee P? \rangle P) \wedge (empty \vee [(Q?; skip)^*] more) \ .$$

The three left-formulas of X are $(P?; Q?)$, $(Q?^* \vee P?)$ and $(Q?; skip)^*$.

DEFINITION 38 (THE SET OF FORMULAS FL_V). — *For any set V of propositional variables, let FL_V denote the subset of* FL *in which all left-formulas are in FE_V. Variables which are not in V can occur within FL_V formulas but not in left-formulas.*

REMARKS 39. — The conventional temporal operators $\bigcirc$ and $\Diamond$ which are primitives in FL can actually be expressed as instances of FL-chop:

$$\models \bigcirc X \equiv \langle skip \rangle X \qquad \models \Diamond X \equiv \langle skip^* \rangle X.$$

These equivalences are found in the FL axiom system presented later in Table 2 as Axioms **FL2** and **FL3**, respectively.

In spite of FL being a proper subset of PITL, the subset of FL_V in which *all* variables are in V (i.e., $\mathrm{FL}_V \cap \mathrm{PITL}_V$) defines exactly the languages in Fusion_V:

LEMMA 40. — *For any language L, the following are equivalent:*

(a) *L is in* Fusion_V.

(b) *L is definable in* FL_V *by some* FL_V *formula having* all *variables in V.*

PROOF. — The implication (b)⇒(a) is the simpler of the two so we consider it first. The previous Lemma 32 ensures any language definable in PITL_V is in Fusion_V. Therefore, since the formula is in PITL_V, the associated language is in Fusion_V.

Let us now establish the implication (a)⇒(b). If L is a language in Fusion_V then Lemma 34 ensures that there exists some FE_V formula E which expresses L, i.e., $\mathcal{L}_V^+(E) = L$. Therefore, the semantically equivalent FL_V formula $\langle E \rangle empty$ also expresses L. This uses the previously noted fact that PTL operator $empty$ is expressible in FL_V. ■

The Star Height and Exterior Height of Formulas in FE *and* FL

Two measures of formula complexity called *star height* and *exterior height* are introduced to assist in an inductive proof of relative completeness for all FL_V formulas.

DEFINITION 41 (STAR HEIGHT). — *The* star height *of an* FE *formula E is the maximum number of nested chop-stars in E. The star height of an* FL *formula X is*

the maximum of the star heights of X's left-formulas. We denote these as $sh(E)$ and $sh'(X)$, respectively.

DEFINITION 42 (EXTERIOR HEIGHT). — *The* exterior height *of an* FE *formula E is defined relative to some number $n \geq 0$ and measures the nesting of constructs in E with star height at least n. This is denoted as $eh_n(E)$ and precisely defined as follows:*

– If $sh(E) < n$, then $eh_n(E) = 0$.

– Otherwise:

- $eh_n(w?) = eh_n(skip) = 1$ *(only used for $n = 0$)*
- $eh_n(F \vee G) = eh_n(F;G) = \max(eh_n(F), eh_n(G)) + 1$
- $eh_n(F^*) = eh_n(F) + 1$.

The exterior height of an FL *formula relative to some star height n is the* maximum *of the exterior heights of its left-formulas relative to n. This is denoted by $eh'_n(X)$.*

We currently prefer to keep sh and eh separate from sh' and eh' since some formulas such as $skip$ and $skip \vee skip$ are both in FE and FL and might be a source of confusion here.

The following subsets of FE_V formulas and FL_V formulas based on star height together with the previously analysed characterisation of chop-star assist in inductively proving deductive completeness for FL_V:

DEFINITION 43 (THE SETS $\mathrm{FE}_V^{0,0}, \mathrm{FE}_V^{0,1}, \ldots, \mathrm{FE}_V^{n,m}, \ldots$). — *For any $m, n \geq 0$, let $\mathrm{FE}_V^{n,m}$ denote the set of all FE_V formulas having maximum star height n and maximum exterior height m relative to n. In other words, an FE_V formula E is in $\mathrm{FE}_V^{n,m}$ iff $sh(E) \leq n$ and $eh_n(E) \leq m$.*

DEFINITION 44 (THE SETS $\mathrm{FL}_V^{0,0}, \mathrm{FL}_V^{0,1}, \ldots, \mathrm{FL}_V^{n,m}, \ldots$). — *For any $m, n \geq 0$, let $\mathrm{FL}_V^{n,m}$ denote the set of all FL_V formulas having maximum star height n and maximum exterior height m relative to n. In other words, an FL_V formula X is in $\mathrm{FL}_V^{n,m}$ iff $sh'(X) \leq n$ and $eh'_n(X) \leq m$.*

9. Axiom System for FL

We now look at the FL axiom system given in Table 2 and later prove that it is complete in Theorem 71. The FL axiom system is designed to provide a way to compose and decompose the left and right sides of FL-chop constructs and to express some useful relations concerning FL formulas similar to those found in the PITL axiom system in Table 1. Natural restrictions imposed by the use of fusion expressions in the FL syntax contribute to some of the variation from the PITL axioms. The FL axiom system, like the one for PITL, permits an embedding of the PTL axiom system.

Table 2. *Axiom system for* FL

Taut	$\vdash$ All FL tautologies	**FL7**	$\vdash \langle E\rangle(X \vee Y) \supset \langle E\rangle X \vee \langle E\rangle Y$
FL2	$\vdash \bigcirc X \equiv \langle skip\rangle X$	**FL8**	$\vdash \langle E^*\rangle X \equiv X \vee \langle E; E^*\rangle X$
FL3	$\vdash \Diamond X \equiv \langle skip^*\rangle X$	**FL9**	$\vdash \Box(X \supset Y) \supset \langle E\rangle X \supset \langle E\rangle Y$
FL4	$\vdash \langle w?\rangle X \equiv w \wedge X$	**FL10**	$\vdash \bigcirc X \supset \circledcirc X$
FL5	$\vdash \langle E \vee F\rangle X \equiv \langle E\rangle X \vee \langle F\rangle X$	**FL11**	$\vdash X \wedge \Box(X \supset \circledcirc X) \supset \Box X$
FL6	$\vdash \langle E; F\rangle X \equiv \langle E\rangle\langle F\rangle X$	**FL12**	$\vdash \Diamond empty$

MP $\vdash X \supset Y, \quad \vdash X \quad \Rightarrow \quad \vdash Y$ $\qquad$ $\Box$**Gen** $\vdash X \quad \Rightarrow \quad \vdash \Box X$

FInf3 $\vdash \langle E\rangle empty \supset \langle F\rangle empty \quad \Rightarrow \quad \vdash \langle E\rangle X \supset \langle F\rangle X$

FInf4 $\vdash (more \wedge \langle E\rangle empty) \supset \langle F\rangle empty \quad \Rightarrow \quad \vdash \langle E^*\rangle X \supset \langle F^*\rangle X$

DEFINITION 45 (FL THEOREMHOOD AND CONSISTENCY). — *An formula X is called an* FL theorem*, denoted $\vdash_{\mathrm{FL}} X$, if X is in* FL *and there exists a sequence of* FL*-deductions which lead to X and only involve formulas in* FL*.*

A formula X is called FL-consistent *if it is in* FL *and its negation is not an* FL *theorem.*

LEMMA 46 (SOUNDNESS OF THE FL AXIOM SYSTEM). — *The* FL *axiom system is sound, that is, any formula which is a theorem is also valid.*

PROOF. — As in Lemma 5 concerning the soundness of the PITL axiom system, the proof here involves induction on the length of an FL theorem's finite proof in the axiom system. The next Lemma 47 proves soundness of Inference Rule **FInf4**. We omit the remaining details. ■

LEMMA 47 (SOUNDNESS OF INFERENCE RULE **FInf4**). — *Suppose for two* FE *formulas E and F the following implication holds:*

$$\models \quad more \wedge \langle E\rangle empty \quad \supset \quad \langle F\rangle empty \ . \tag{1}$$

Then for any FL *formula X, the following implication is also valid:*

$$\models \quad \langle E^*\rangle X \quad \supset \quad \langle F^*\rangle X \ . \tag{2}$$

PROOF. — We first ensure that any interval satisfying E^* also satisfies F^* and prove this by induction on interval length. In the case of an empty interval, F^* is trivially true for any F. A nonempty interval satisfying E^* can be split into two subintervals in which the first is also nonempty and satisfies E and the second satisfies E^*. By assumption (1), the first subinterval also satisfies F. Induction in interval length ensures that the second subinterval satisfies F^*. Therefore, the overall interval satisfies the formula $F; F^*$ and hence also F^*. This readily leads to our goal, namely, the validity of the implication (2). ■

Observe that for each value of V, the axiom system provides a means of deducing theorems only involving formulas in FL_V. Thus for each possible V, the associated

set FL_V can be viewed as a logic in its own right. Here is a formal definition of deducibility and consistency along these lines:

DEFINITION 48 (FL_V THEOREMHOOD AND CONSISTENCY). — *An formula X is called an* FL_V theorem, *denoted* $\vdash_{\mathrm{FL}_V} X$, *if X is in* FL_V *and there exists a sequence of* FL*-deductions which lead to X and only involve formulas in* FL_V.

A formula X is called FL_V-consistent *if it is in* FL_V *and its negation is not an* FL_V *theorem.*

LEMMA 49 (SUBSTITUTION INSTANCES OF FL_V THEOREMS). — *Let X be an* FL_V *theorem, $Y_1, \ldots, Y_n$ be* FL_V *formulas and $R_1, \ldots, R_n$ be variables not in V. Then the substitution instance $X^{Y_1,\ldots,Y_n}_{R_1,\ldots,R_n}$ is itself an* FL_V *theorem.*

PROOF. — Each axiom and application of an inference rule in X's proof can have the original formulas involved replaced by ones in which the variables $R_1, \ldots, R_n$ are simultaneously replaced by $Y_1, \ldots, Y_n$. Due to the assumption that $R_1, \ldots, R_n$ are not variables in V, the resulting formulas are well-formed FL_V ones. Therefore, since none of the variables $R_1, \ldots, R_n$ occur in any fusion expressions in the proof of X, the proof of the theoremhood of $X^{Y_1,\ldots,Y_n}_{R_1,\ldots,R_n}$ remains deducible and indeed has the same length as the original one for X. ■

10. PTL Axiom System

We now present a complete axiom system for PTL which can be embedded in the FL axiom system. This will then enable us to show that all PTL formulas which are valid for finite time can be deduced as FL theorems. The PTL axiom system considered here and shown in Table 3 is derived from another similar PTL axiom system DX proposed by Pnueli [PNU 77]. Gabbay et al. [GAB 80] showed that DX is complete. Pnueli's original system uses strong versions of $\Diamond$ and $\Box$ which do not examine the current state. In addition, Pnueli's system only deals with infinite time. However, Gabbay et al. [GAB 80] also include a variant system called $\mathrm{D}^0\mathrm{X}$ based on the conventional $\Diamond$ and $\Box$ operators which examine the current state. The slightly modified version presented here does this as well and also permits both finite and infinite time. Finite time is essential in our completeness proofs for FL and PITL. Optionally, an axiom such as $\Diamond\, empty$ can added to restrict time to being finite.

Table 3. *Modified version of Pnueli's complete* PTL *axiom system* DX

Axioms:

A1	$\vdash \Box(X \supset Y) \supset \Box X \supset \Box Y$
A2	$\vdash \bigcirc X \supset \textcircled{w} X$
A3	$\vdash \bigcirc(X \supset Y) \supset \bigcirc X \supset \bigcirc Y$
A4	$\vdash \Box X \supset X \wedge \textcircled{w} \Box X$
A5	$\vdash \Box(X \supset \textcircled{w} X) \supset X \supset \Box X$

Inference rules:

R1	If X is a tautology, then $\vdash X$
R2	If $\vdash X \supset Y$ and $\vdash X$, then $\vdash Y$
R3	If $\vdash X$, then $\vdash \Box X$

We denote the validity and theoremhood of a PTL formula X within the PTL framework by $\models_{\mathrm{PTL}} X$ and $\vdash_{\mathrm{PTL}} X$, respectively.

LEMMA 50 (SOUNDNESS OF THE PTL AXIOM SYSTEM). — *If a* PTL *formula X is derivable as a theorem of the* PTL *axiom system, it is also valid. In other words, if $\vdash_{\mathrm{PTL}} X$, then $\models_{\mathrm{PTL}} X$.*

THEOREM 51 (COMPLETENESS OF THE PTL AXIOM SYSTEM). — *Any valid* PTL *formula is deducible as a theorem in the* PTL *axiom system in Table 3.*

PROOF. — This is established by us in [MOS 04a]. ■

10.1. *Deducibility of* PTL *Theorems within the* FL *Axiom System*

It is necessary to show that the PTL axiom system found in Table 3 can be embedded in the FL axiom system given in Table 2. This in useful in its own right since for any V, all PTL theorems are FL_V theorems as well. In addition, we later prove completeness for FL_V relative to PTL. This combined with the completeness of the PTL axiom system then yields completeness for formulas in FL_V and indeed all of FL. As part of the proof, we need to establish that if a PTL formula X is FL-consistent, then some interval satisfies the formula $X \wedge \mathit{finite}$. This interval can then serve as an FL model for X.

LEMMA 52 (EVERY PTL THEOREM IS AN FL THEOREM). — *For any* PTL *formula X, if X is a* PTL *theorem, then it is also an* FL *theorem, i.e., if $\vdash_{\mathrm{PTL}} X$, then $\vdash_{\mathrm{FL}} X$ and also for any V, $\vdash_{\mathrm{FL}_V} X$ holds. Moreover, the deductions do not require the* FL *Inference Rule* **FInf4**.

PROOF. — All the PTL axioms and inference rules can be embedded in the FL axiom system. Therefore, the deductions in the proof of $\vdash_{\mathrm{PTL}} X$ can be mimicked to obtain $\vdash_{\mathrm{FL}} X$. Details of the proof can be found in Appendix B. The reasoning only involves FL formulas in which the fusion expressions do not contain any variables. Therefore, any such FL theorem is also an FL_V theorem for any V. ■

LEMMA 53 (COMPLETENESS OF THE FL AXIOM SYSTEM FOR PTL WITH FINITE TIME). — *Let X be a* PTL *formula which is valid for finite time (i.e., $\models_{\mathrm{PTL}} \mathit{finite} \supset X$). Then $\vdash_{\mathrm{FL}} X$ holds and in addition, for any V, $\vdash_{\mathrm{FL}_V} X$ holds. Moreover, the deductions do not require the* FL *Inference Rule* **FInf4**.

PROOF. — Suppose $\models_{\mathrm{PTL}} \mathit{finite} \supset X$ holds. Then by the completeness of the PTL axiom system, we can deduce the PTL theorem $\vdash_{\mathrm{PTL}} \mathit{finite} \supset X$ and hence by Lemma 52, the FL theorem $\vdash_{\mathrm{FL}} \mathit{finite} \supset X$. This lemma does not require the FL Inference Rule **FInf4**. Now we also can readily deduce the FL theorem $\vdash_{\mathrm{FL}} \mathit{finite}$ from the FL Axiom **FL12** together with the definition of finite as $\Diamond\, \mathit{empty}$. Modus ponens then yields desired FL theorem $\vdash_{\mathrm{FL}} X$. The embedding of the PTL proof of X in FL contains no variables within the scope of FE formulas. Therefore X is also an FL_V theorem for any V. ■

LEMMA 54 (SUBSTITUTION INSTANCES OF VALID PTL FORMULAS IN FL_V). — *Let X be some* PTL *formula which is valid for finite time, $Y_1, \ldots, Y_n$ be FL_V formulas and $R_1, \ldots, R_n$ be variables not in V. Then the substitution instance $X^{Y_1,\ldots,Y_n}_{R_1,\ldots,R_n}$ is itself an FL_V theorem.*

PROOF. — If X is valid for finite time then Lemma 53 concerning completeness for PTL formulas in the FL axiom system ensures that X is deducible as an FL_V theorem. Lemma 49 then ensures that $X^{Y_1,\ldots,Y_n}_{R_1,\ldots,R_n}$ is an FL_V theorem. ■

10.2. *Substitution within* FL_V *Formulas*

We now present some lemmas concerning substitution within FL_V formulas which are in addition to the earlier Lemma 49. The proofs include PTL-based reasoning so it seems natural and convenient to present the lemmas after our discussion about FL_V theoremhood for substitution instances of PTL formulas which are themselves valid in finite time.

LEMMA 55 (THE TEMPORAL OPERATOR $\Box$ AND FL_V THEOREMHOOD). — *Let X be an arbitrary FL_V formula and R be a propositional variable not in V and further let Y be an FL_V formula not containing R. Then the following implication is deducible as an FL_V theorem:*

$$\vdash_{\mathrm{FL}_V} \Box(R \equiv Y) \quad \supset \quad X \equiv X^Y_R \ .$$

PROOF. — Induction can be done on X's syntax. ■

LEMMA 56 (FL_V THEOREMHOOD OF SUBSTITUTION INSTANCES OF DEDUCIBLE EQUIVALENCES). — *Let Y and Z be deductively equivalent FL_V formulas (i.e., $\vdash_{\mathrm{FL}_V} Y \equiv Z$) and further let X be an arbitrary FL_V formula and R be a propositional variable not in V and not occurring in Y or Z. Then the formulas X^Y_R and X^Z_R are deductively equivalent within FL_V, i.e., $\vdash_{\mathrm{FL}_V} X^Y_R \equiv X^Z_R$.*

PROOF. — We use Lemma 55 to deduce the following implication as an FL_V theorem:

$$\vdash_{\mathrm{FL}_V} \Box(R \equiv Z) \quad \supset \quad X \equiv X^Z_R \ .$$

Lemma 49 is then invoked to substitute Y into R:

$$\vdash_{\mathrm{FL}_V} \Box(Y \equiv Z) \quad \supset \quad X^Y_R \equiv X^Z_R \ . \tag{3}$$

In addition, the deductive equivalence of Y and Z together with the FL_V Inference Rule $\Box$**Gen** ensures the FL_V theoremhood of the next formula:

$$\vdash \Box(Y \equiv Z) \ . \tag{4}$$

The combination of formulas (3) and (4) together with modus ponens yields our goal of the FL_V theoremhood of the equivalence $X^Y_R \equiv X^Z_R$. ■

We also make use of $\Box$ to assist in obtaining relative completeness for substitution instances:

LEMMA 57 (THE TEMPORAL OPERATOR $\Box$ AND RELATIVE COMPLETENESS IN FL_V). — *Let X be an arbitrary FL_V formula and R be a propositional variable not in V and further let Y be an FL_V formula not containing R. Then completeness holds for the substitution instance X_R^Y relative to the following conjunction:*

$$\Box(R \equiv Y) \ \wedge \ X \ . \tag{5}$$

PROOF. — We can show the following two properties:

(a) If X_R^Y is FL_V-consistent then so is (5).

(b) If (5) is satisfiable then so is X_R^Y.

Proof of (a): Suppose on the contrary that formula (5) is not FL_V-consistent. Then $\Box(R \equiv Y) \supset \neg X$ is an FL_V theorem. By Lemma 49 so is the substitution instance $\Box(Y \equiv Y) \supset \neg X_R^Y$ from which we readily obtain that $\neg X_R^Y$ is an FL_V theorem. Therefore X_R^Y is not FL_V-consistent.

Proof of (b): Lemma 55 and the soundness of the FL axiom system (Lemma 46) yield the validity of the next implication:

$$\models \quad \Box(R \equiv Y) \quad \supset \quad X \equiv X_R^Y \ .$$

Therefore any model for $\Box(R \equiv Y) \wedge X$ itself satisfies X_R^Y.

Parts (a) and (b) together with Corollary 14 ensure FL_V-completeness for X_R^Y relative to the formula (5). ■

11. FE Formula Behaviour in Empty and Nonempty Intervals

Performing induction over time on an FL formula $\langle E^* \rangle X$ can be tricky. One challenge is that E might be true on some empty intervals and therefore when $\langle E^* \rangle X$ is unwound into $\langle E; E^* \rangle X$ by means of Axiom **FL8**, the first E can collapse, thereby preventing any advance to a strictly later state. Because of this difficulty, a function c is introduced which for any $n \geq 0$ transforms an arbitrary formula E in $\mathrm{FE}_V^{n,0}$ into another formula $c(E)$ also in $\mathrm{FE}_V^{n,0}$. This new formula $c(E)$ captures E's behaviour in nonempty intervals and the PITL equivalence $c(E) \equiv (E \wedge \mathit{more})$ is valid. Unlike E, $c(E)$ cannot collapse. Therefore it facilitates dealing with the reduction of chop-star instances since if E is arbitrary, the FL_V formula $\langle c(E)^* \rangle X$ is easier to unwind than the semantically equivalent formula $\langle E^* \rangle X$. The fact that the star height of $c(E)$ is no greater than E's turns out to greatly assist us in obtaining completeness for FL_V when we do induction on the star height in formulas.

REMARKS 58. — Some readers may have trouble understanding that individual iterations of chop-star can occur in empty intervals. As a result, it may be difficult to

accept that the semantic equivalence of $c(E)$ and $E \wedge more$ is sufficient to ensure the semantic equivalence of $\langle c(E)^* \rangle X$ and $\langle E^* \rangle X$. Nevertheless, this equivalence indeed holds even though $c(E)^*$ avoids empty iterations whereas E^* might permit them. The proof of Lemma 47 concerning the soundness of the FL Inference Rule **FInf4** for chop-star explains why for any two FE formulas E and F with the valid FL implication $\models more \wedge \langle E \rangle empty \supset \langle F \rangle empty$, the FE formula E^* implies F^*. Consequently, any empty iterations possibly generated by E can be ignored. When F refers to a formula such as $c(E)$, we have the following stronger assumption:

$$\models \quad \langle F \rangle empty \quad \equiv \quad \langle E \rangle empty \wedge more \ .$$

In such a case, it is even easier to see that the converse implication from F^* to E^* also holds. Therefore E^* and F^* are semantically equivalent. The later Subsect. 14.3 contains a related analysis concerning chop-star in the context of PITL.
END OF REMARKS 58.

Below is the definition of c:

DEFINITION 59 (THE FUNCTION c). — *For any* FE *formula* E*, define* $c(E)$ *in the following way:*

E	$c(E)$
$w?$	$false?$
$F \vee G$	$c(F) \vee c(G)$
$skip$	$skip$
$F;G$	$c(F);G \ \vee \ F;c(G)$
F^*	$c(F);F^*$

LEMMA 60 (STAR HEIGHT OF $c(E)$ EQUALS STAR HEIGHT OF E). — *For any* FE_V *formula* E*, we have* $sh(c(E)) = sh(E)$ *and* $c(E)$ *also in* FE_V*. Also for any* $n \geq 0$*, if* E *is in* $\mathrm{FE}_V^{n,0}$*, then the formula* $c(E)$ *is also in* $\mathrm{FE}_V^{n,0}$*.*

PROOF. — In order to show that $sh(c(E)) = sh(E)$ and $c(E)$ is in FE_V, we do induction on E's syntax and observe that no case in $c(E)$'s definition adds layers of chop-stars or new variables. Now E is in $\mathrm{FE}_V^{n,0}$ iff it is in FE_V and $sh(E) < n$. Therefore $c(E)$ is likewise in $\mathrm{FE}_V^{n,0}$. ■

We now formally define the notion of *nonempty formulas*:

DEFINITION 61 (NONEMPTY FE FORMULAS). — *An* FE *formula* E *is said to be* nonempty *if it is not true on empty intervals and hence the next two implications are valid:*

$$\models \quad E \quad \supset \quad more$$

$$\models \quad \langle E \rangle empty \quad \supset \quad more \ .$$

LEMMA 62 ($c(E)$ IS NONEMPTY). — *For any* FE *formula* E *and* FL_V *formula* X, *the formula* $\langle c(E)\rangle X$ *is nonempty and hence the* FL *implication* $\langle c(E)\rangle empty \supset more$ *is valid.*

PROOF. — This follows by induction on E's syntax. ■

LEMMA 63 (SEMANTIC EQUIVALENCE FOR $c(E)$). — *For any* FE *formula* E, *the following* FL *equivalence is valid:*

$$\models \quad \langle c(E)\rangle empty \quad \equiv \quad \langle E\rangle empty \ \wedge \ more \ . \tag{6}$$

PROOF. — We do induction on E's syntax to prove the validity of the previously mentioned PITL formula $c(E) \equiv (E \wedge more)$. This is semantically equivalent to the equivalence in our goal (6). ■

LEMMA 64 (SEMANTIC EQUIVALENCE OF E^* AND $c(E)^*$). — *For any* FE *formula* E, *the* FE *formulas* E^* *and* $c(E)^*$ *are semantically equivalent.*

PROOF. — By Lemma 63 the only difference between the behaviour of E and $c(E)$ concerns empty intervals. Such intervals can be safely ignored in individual iterations taking place in E^* and $c(E)^*$ so these two formulas have identical semantics. ■

12. Indirect Characterisation of Chop-Star

We will make use of the hierarchy of sets of FL_V formulas in Definition 44 based on star height and relative exterior height. For any $m, n \geq 0$, there is a corresponding set denoted as $\mathrm{FL}_V^{n,m}$. Later we want to show in Lemma 68 that completeness holds the set $\mathrm{FL}_V^{n,m+1}$ relative to $\mathrm{FL}_V^{n,m}$. This requires some means of relating formulas with exterior height $m+1$ relative to n to others with exterior height m relative to n. In preparation for doing this, we now describe an important way of eliminating individual chop-stars from formulas by indirectly mimicking chop-star behaviour with the aid of an auxiliary variable. Consider the analogous situation in conventional PTL with the operator *until*. There it is possibly to show that completeness for a formula containing instances of *until* holds relative to another *until*-free one which in effect mimics each instance of *until* by means of an extra variable. We can apply a similar technique in FL to reasoning about chop-star in finite time.

The previous Lemma 64 ensures that the FE formulas E^* and $c(E)^*$ have identical semantics. This can be exploited to indirectly characterise E^* using $c(E)$. Let R be a propositional variable, E be an FE formula and X be an FL formula. We later establish in Theorem 66 that under suitable assumptions the formula $\Box(R \equiv \langle E^*\rangle X)$ is deducibly equivalent to the formula $\Box\big(R \ \equiv \ (X \vee \langle c(E)\rangle R)\big)$. Now the number of FE formulas with the same star height as E^* in the formula $\Box\big(R \ \equiv \ (X \vee \langle c(E)\rangle R)\big)$ is one less than in the semantically equivalent original formula $\Box(R \equiv \langle E^*\rangle X)$ since the earlier Lemma 60 ensures that $c(E)$ has the same star height as E (i.e., $sh(c(E)) = sh(E)$). Later in Lemma 68 in Sect. 13 we use Theorem 66 to prove a kind of relative

completeness by reducing certain useful FL_V formulas containing chop-star to others with strictly lower star height.

LEMMA 65 (DEDUCIBLE UNWINDING OF E^* USING $c(E)$). — *Suppose E is an $\mathrm{FE}_V^{n,0}$ formula and X is some FL_V formula. If completeness holds for $\mathrm{FL}_V^{n,0}$, then the following formula is an FL_V theorem:*

$$\vdash_{\mathrm{FL}_V} \langle E^* \rangle X \quad \equiv \quad X \ \vee \ \langle c(E) \rangle \langle E^* \rangle X \ . \tag{7}$$

PROOF. — If E is in $\mathrm{FE}_V^{n,0}$, then Lemma 60 ensures that $c(E)$ is also in $\mathrm{FE}_V^{n,0}$. Lemma 63 and simple semantic reasoning yield the next two valid implications:

$$\begin{aligned} &\models \quad more \ \wedge \ \langle E \rangle empty \quad \supset \quad \langle c(E) \rangle empty \\ &\models \quad more \ \wedge \ \langle c(E) \rangle empty \quad \supset \quad \langle E \rangle empty \ . \end{aligned}$$

The assumption of completeness for $\mathrm{FL}_V^{n,0}$ guarantees both are FL_V theorems:

$$\begin{aligned} &\vdash_{\mathrm{FL}_V} \ more \ \wedge \ \langle E \rangle empty \quad \supset \quad \langle c(E) \rangle empty \\ &\vdash_{\mathrm{FL}_V} \ more \ \wedge \ \langle c(E) \rangle empty \quad \supset \quad \langle E \rangle empty \ . \end{aligned}$$

These with two respective applications of Inference Rule **FInf4** yield the following deducible implications:

$$\begin{aligned} &\vdash_{\mathrm{FL}_V} \ \langle E^* \rangle empty \quad \supset \quad \langle c(E)^* \rangle empty \\ &\vdash_{\mathrm{FL}_V} \ \langle c(E)^* \rangle empty \quad \supset \quad \langle E^* \rangle empty \ . \end{aligned}$$

Both implications are used in two respective applications of Inference Rule **FInf3** together with some propositional reasoning to deduce the following equivalence:

$$\vdash_{\mathrm{FL}_V} \ \langle E^* \rangle X \quad \equiv \quad \langle c(E)^* \rangle X \ . \tag{8}$$

The right side can be unwound by means of Axiom **FL8** and then Axiom **FL6**:

$$\vdash_{\mathrm{FL}_V} \ \langle E^* \rangle X \quad \equiv \quad X \vee \langle c(E) \rangle \langle c(E)^* \rangle X \ . \tag{9}$$

These last two equivalences (8) and (9) combined with Lemma 56 permit replacing $\langle c(E)^* \rangle X$ with $\langle E^* \rangle X$ in (9) to ensure our goal (7). ■

THEOREM 66 (DEDUCIBLE INDIRECT CHARACTERISATION OF CHOP-STAR). — *Suppose R is a propositional variable, E is in $\mathrm{FE}_V^{n,0}$ and X is in $\mathrm{FL}_V^{n,m}$. Then if completeness holds for $\mathrm{FL}_V^{n,m}$, the following equivalence is an FL_V theorem:*

$$\vdash_{\mathrm{FL}_V} \ \Box(R \equiv \langle E^* \rangle X) \quad \equiv \quad \Box(R \ \equiv \ (X \ \vee \ \langle c(E) \rangle R)) \ . \tag{10}$$

PROOF. — Let P be some propositional variable which is distinct from R, not in V and not occurring in X. For any nonempty FE formula F, the following implication can be shown to be valid by induction on interval length:

$$\models \quad \Box(P \;\equiv\; (X \vee \langle F\rangle P)) \;\wedge\; \Box(R \;\equiv\; (X \vee \langle F\rangle R)) \quad \supset \quad \Box(R \equiv P) \;.$$

From this we can readily obtain the validity of the next implication:

$$\models \quad \Box(P \;\equiv\; (X \vee \langle F\rangle P)) \quad \supset \quad \Box(R \equiv P) \;\equiv\; \Box(R \;\equiv\; (X \vee \langle F\rangle R)) \;. \tag{11}$$

Lemma 62 guarantees that $c(E)$ is a nonempty formula. We can therefore replace F by $c(E)$ in (11) to obtain the following implication:

$$\begin{aligned} \models \quad & \Box(P \;\equiv\; (X \vee \langle c(E)\rangle P)) \\ & \supset \quad \Box(R \equiv P) \;\equiv\; \Box(R \;\equiv\; (X \vee \langle c(E)\rangle R)) \;. \end{aligned} \tag{12}$$

The assumption that E is in $\mathrm{FE}_V^{n,0}$ and Lemma 60 ensure that $c(E)$ is also in $\mathrm{FE}_V^{n,0}$. In addition, X is assumed to be in $\mathrm{FL}_V^{n,m}$. Therefore formula (12) is in $\mathrm{FL}_V^{n,m}$ and by the assumption of completeness for $\mathrm{FL}_V^{n,m}$ is an FL_V theorem:

$$\begin{aligned} \vdash_{\mathrm{FL}_V} \; & \Box\big(P \;\equiv\; (X \vee \langle c(E)\rangle P)\big) \\ & \supset \quad \Box(R \equiv P) \;\equiv\; \Box(R \;\equiv\; (X \vee \langle c(E)\rangle R)) \;. \end{aligned}$$

We invoke Lemma 49 to substitute the formula $\langle E^*\rangle X$ into P and obtain the following implication:

$$\begin{aligned} \vdash_{\mathrm{FL}_V} \; & \Box\big(\langle E^*\rangle X \;\equiv\; (X \vee \langle c(E)\rangle\langle E^*\rangle X)\big) \\ & \supset \quad \Box(R \equiv \langle E^*\rangle X) \;\equiv\; \Box(R \;\equiv\; (X \vee \langle c(E)\rangle R)) \;. \end{aligned}$$

Now the deducible equivalence (7) established in Lemma 65 combined with the Inference Rule $\Box$**Gen** yields the antecedent of this implication:

$$\vdash_{\mathrm{FL}_V} \; \Box\big(\langle E^*\rangle X \;\equiv\; (X \vee \langle c(E)\rangle\langle E^*\rangle X)\big) \;.$$

Consequently, we can use modus ponens to arrive at our goal (10). ■

13. Proof of Completeness for the FL Axiom System

Axiomatic completeness for the FL axiom system is established by first taking some arbitrary finite set of variables V and ensuring completeness for FL_V formulas. We does this by inductively proving completeness for a hierarchy of subsets of FL_V formulas. Let us recall the diagrammatic summary of the interrelationship of the completeness proofs presented earlier in Subsect. 1.2:

$$\begin{aligned} \mathrm{PTL} \xrightarrow{67} \mathrm{FL}_V^{0,0} \longrightarrow \cdots \longrightarrow \mathrm{FL}_V^{n,m} \xrightarrow{68} \mathrm{FL}_V^{n,m+1} \\ \longrightarrow \cdots \xrightarrow{69} \mathrm{FL}_V^{n+1,0} \longrightarrow \cdots \xrightarrow{70} \mathrm{FL}_V \xrightarrow{71} \mathrm{FL} \;. \end{aligned}$$

Each numbered arrow corresponds to a lemma or theorem now presented.

LEMMA 67 (COMPLETENESS FOR $\mathrm{FL}_V^{0,0}$). — *Completeness holds for* $\mathrm{FL}_V^{0,0}$.

PROOF. — Formulas in $\mathrm{FL}_V^{0,0}$ contain no FL-chop constructs and are hence $\mathrm{FL}_V^{0,0}$ is identical to the set of all PTL formulas. Now Lemma 53 establishes completeness for PTL with finite time in the FL axiom system and hence also ensures completeness for $\mathrm{FL}_V^{0,0}$ in the FL axiom system. ■

LEMMA 68 (RELATIVE COMPLETENESS FOR $\mathrm{FL}_V^{n,m+1}$). — *For any* $m, n \geq 0$, *completeness for* $\mathrm{FL}_V^{n,m+1}$ *holds relative to* $\mathrm{FL}_V^{n,m}$.

PROOF. — Let X refer some formula in $\mathrm{FL}_V^{n,m+1}$ and let k be the number of left-formulas in X with exterior height $m+1$ relative to star height n. We do induction on k to show completeness holds for X relative to $\mathrm{FL}_V^{n,m}$.

Base case for $k = 0$: This is trivial since X itself is in $\mathrm{FL}_V^{n,m}$.

Induction step from k *to* $k+1$: Suppose X contains exactly $k+1$ left-formulas having exterior height $m+1$ relative to n. Let R be a propositional variable not in V and not occurring in X. We can express X as $Y_R^{\langle E \rangle Y'}$, where the following conditions hold:

– The formula Y is in $\mathrm{FL}_V^{n,m+1}$.

– Only k left-formulas occur in Y with exterior height $m+1$ relative to n.

– The formula E is in $\mathrm{FE}^{n,m+1}$.

– The formula Y' is in $\mathrm{FL}_V^{n,m}$.

Lemma 57 ensures that completeness holds for $Y_R^{\langle E \rangle Y'}$ relative to the next conjunction which is also in $\mathrm{FL}_V^{n,m+1}$:

$$\Box(R \equiv \langle E \rangle Y') \wedge Y \quad . \tag{13}$$

Therefore completeness also holds for X relative to this.

We now construct a new formula $\Box(R \equiv Z) \wedge Y$ which is deducibly equivalent to (13) and in $\mathrm{FL}_V^{n,m}$. If E's outermost construct is not chop-star, we can simply determine the formula Z based on E's outermost operator:

E	$\langle E \rangle Y'$	Z
$w?$	$\langle w? \rangle Y'$	$w \wedge Y'$
$F \vee G$	$\langle F \vee G \rangle Y'$	$\langle F \rangle Y' \vee \langle G \rangle Y'$
$skip$	$\langle skip \rangle Y'$	$\bigcirc Y'$
$F;G$	$\langle F;G \rangle Y'$	$\langle F \rangle \langle G \rangle Y'$.

In each of these cases $\langle E \rangle Y'$ is easily shown to be deducibly equivalent to Z by means of FL axioms. Therefore by Lemma 56 the formula $\Box(R \equiv Z) \wedge Y$ is indeed deducibly equivalent to (13).

The remaining case where E's outermost operator is chop-star is the only non-trivial one. Here we let Z be the formula $Y' \vee \langle c(E)\rangle R$. The overall formula $\Box(R \equiv Z) \wedge Y$ is deducibly equivalent to (13). This is because Theorem 66 and the inductive assumption of completeness for $\mathrm{FL}_V^{n,m}$ ensure that $\Box(R \equiv \langle E^*\rangle Y')$ and $\Box(R \equiv Z)$ are themselves deducibly equivalent.

In all of the cases it follows that completeness for X holds relative to $\Box(R \equiv Z) \wedge Y$. Also, $\Box(R \equiv Z) \wedge Y$ has only k left-formulas with exterior height $m+1$ relative to n. Therefore induction on k yields that completeness holds for $\Box(R \equiv Z) \wedge Y$ relative to $\mathrm{FL}_V^{m,n}$. Lemma 12 then ensures that completeness also holds for X relative to $\mathrm{FL}_V^{m,n}$. ■

LEMMA 69 (RELATIVE COMPLETENESS FOR $\mathrm{FL}_V^{n+1,0}$). — *For any $n \geq 0$, completeness for $\mathrm{FL}_V^{n+1,0}$ holds relative to the union of the countably infinite sets $\mathrm{FL}_V^{n,0}$, $\mathrm{FL}_V^{n,1}, \ldots$.*

PROOF. — This readily follows from the fact that the set $\mathrm{FL}_V^{n+1,0}$ itself equals the union of the sequence of sets $\mathrm{FL}_V^{n,0}$, $\mathrm{FL}_V^{n,1}, \ldots$. ■

THEOREM 70 (COMPLETENESS OF FL_V IN THE FL AXIOM SYSTEM). — *Every valid FL_V formula is deducible as an FL_V theorem in the* FL *axiom system.*

PROOF. — It is not hard to see that the set of formulas FL_V equals the union of the sequence of countably infinite sets $\mathrm{FL}_V^{0,0}$, $\mathrm{FL}_V^{0,1}$, ..., $\mathrm{FL}_V^{n,m}$, ... We prove completeness for FL_V by inductively showing that for any $m, n \geq 0$, completeness holds $\mathrm{FL}_V^{n,m}$. This is done using lexicographical ordering of the pair (n, m):

Base case for $\mathrm{FL}_V^{0,0}$: This is established by the previous Lemma 67.

Inductive step for $\mathrm{FL}_V^{n,m+1}$: This is a consequence of the earlier Lemma 68.

Inductive step for $\mathrm{FL}_V^{n+1,0}$: This follows from the earlier Lemma 69. ■

THEOREM 71 (COMPLETENESS OF THE FL AXIOM SYSTEM). — *Every valid* FL *formula is a theorem in the* FL *axiom system.*

PROOF. — Let X be a valid FL formula and V be the set of variables in X. Then X is in FL_V and by Lemma 70 an FL_V theorem and so also an FL theorem. ■

14. Embedding the FL and PTL Axiom Systems in the PITL Axiom System

We first embed the FL axiom system in the PITL one and later ensure that each PITL formula is deductively equivalent to an FL one. The embedding is done in two parts because FL Axiom **FInf4** requires special attention and is only dealt with after the rest of the FL axiom system is considered.

14.1. *Partial Embedding of the* FL *Axiom System in the* PITL *Axiom System*

The next Lemma 72 describes an embedding of most of the FL axiom system in the PITL one. This is sufficient to indirectly embed the PTL axiom system in the PITL one using Lemma 53. Later on, after proving some further properties of chop-star in PITL, in Lemma 78 we can finish the embedding and include the remaining FL_V inference rule **FInf4** not handled by Lemma 72.

LEMMA 72 (PARTIAL EMBEDDING OF FL AXIOM SYSTEM IN PITL AXIOM SYSTEM). — *All* FL *axioms and the three Inference Rules* **MP**, □**Gen** *and* **FInf3** *can be embedded in the* PITL *axiom system.*

PROOF. — We show that each FL axiom and the three mentioned inference rules can be obtained from the PITL axiom system. ■

LEMMA 73 (RESTRICTED SUBSTITUTION OF FL_V THEOREMS IN PITL). — *Let* X *be an* FL_V *theorem deducible without* FL *Inference Rule* **FInf4**. *Also, let* $A_1, \ldots, A_n$ *be* PITL *formulas and* $R_1, \ldots, R_n$ *be variables not in* V. *Then the substitution instance* $X^{A_1,\ldots,A_n}_{R_1,\ldots,R_n}$ *is itself a* PITL *theorem.*

PROOF. — The proof is similar to that for Lemma 49 and uses Lemma 72. ■

We only use this lemma here to obtain substitution instances of PTL theorems in the next Lemma 74. Therefore, the restriction is not a problem.

LEMMA 74 (SUBSTITUTION INSTANCES OF VALID PTL FORMULAS IN PITL). — *Any* PITL *formula which is a substitution instance of a* PTL *formula which is itself valid in finite time is a deducible* PITL *theorem.*

PROOF. — We use Lemmas 53 and 73. ■

14.2. *A Lemma for Restricted Replacement of* PITL *Formulas*

We now present a lemma concerning the replacement of PITL formulas. This Restricted Replacement Lemma permits the replacement of deductively equivalent PITL formulas within a larger PITL formula. However, the lemma cannot deal with replacement in the scope of chop-star. Nevertheless, it is still useful. Later on, a version without this limitation is presented as the PITL Replacement Theorem (Theorem 77). Before that more powerful lemma can be proved, certain PITL theorems concerning the unwinding of chop-star must be established. Part of the reasoning relies on the present restricted replacement lemma although we will not delve into the details.

LEMMA 75 (RESTRICTED REPLACEMENT LEMMA FOR PITL). — *Let* B_1 *and* B_2 *be deducibly equivalent formulas (i.e.,* $\vdash_{\mathrm{PITL}} B_1 \equiv B_2$*). Suppose* A_1 *is an arbitrary formula and the formula* A_2 *is obtained from* A_1 *by replacing within* A_1 *zero or more instances of* B_1 *by* B_2. *If none of these occurrences of* B_1 *are within any chop-star construct, then* A_1 *and* A_2 *are provably equivalent, i.e.,* $\vdash_{\mathrm{PITL}} A_1 \equiv A_2$.

PROOF. — Induction can be done on A_1's syntax with each instance of B_1 regarded as atomic. Lemma 74 and the PITL Axiom **P8** together with the Inference Rules $\Box$**Gen** and $\boxdot$**Gen** and propositional reasoning greatly facilitate the proof. ■

14.3. *Some Properties of Chop-Star in* PITL

The PITL Replacement Theorem (Theorem 77) is a stronger version of Lemma 75 that also handles chop-star and enables us in Lemma 78 to embed the remaining FL Inference Rule **FInf4** in the PITL axiom system. Before presenting it, we need to prove some properties concerning both chop-star and PITL.

As was noted in Section 11 with regard to FL, it is often desirable to be able to assume that individual iterations of a chop-star formula are satisfied in nonempty intervals in order to facilitate induction over time. Indeed, reasoning of this sort is related to the soundness of the PITL Axiom **P11** which only concerns the first nonempty iteration of a chop-star formula. Nonempty iterations are required for proving certain lemmas dealing with substitution into formulas containing chop-star constructs. The next two lemmas are relevant to this:

LEMMA 76. — *For any* PITL *formulas A and B, if the implication $(A \wedge more) \supset B$ is valid, then so is the implication $A^* \supset B^*$.*

PROOF. — We give two different proofs. The first is simpler whereas the second lends itself to being expressed in the axiom system, thus contributing to our ultimate goal of showing deductive completeness.

A direct proof establishes $\models A^* \supset B^*$ by simply deleting all empty iterations in any interval satisfying A^*. The remaining iterations each satisfy $A \wedge more$ and hence also B. Therefore the overall interval satisfies B.

The second proof of $\models A^* \supset B^*$ shows that if the implication does not hold, there must exist some time in the strict future when this situation repeats itself. This can be done by first unwinding the chop-stars using nonempty iterations (as in Axiom **P11**):

$$\begin{aligned} \models \quad & A^* \;\equiv\; empty \vee (A \wedge more); A^* \\ \models \quad & B^* \;\equiv\; empty \vee (B \wedge more); B^* \quad . \end{aligned}$$

We then suitably combine the indicated behaviour of each one:

$$\models \quad A^* \wedge \neg(B^*) \quad \supset \quad \big((A \wedge more); A^*\big) \;\wedge\; \neg\big((B \wedge more); B^*\big) \quad .$$

The assumption $\models (A \wedge more) \supset B$ lets us here replace $B \wedge more$ by $A \wedge more$:

$$\models \quad A^* \wedge \neg(B^*) \quad \supset \quad \big((A \wedge more); A^*\big) \;\wedge\; \neg\big((A \wedge more); B^*\big) \quad .$$

The two chop constructs in the implication's right side are then merged:

$$\models \quad A^* \wedge \neg(B^*) \quad \supset \quad (A \wedge more); \big(A^* \wedge \neg(B^*)\big) \quad .$$

The finiteness of intervals then yields a contradiction. The following valid PTL implication expresses this using some arbitrary propositional variable P:

$$\models_{\mathrm{PTL}} \ \mathit{finite} \wedge \Box(P \supset \bigcirc \Diamond P) \ \supset \ \neg P \ .$$

Below is a variant in the style of an inference rule for an arbitrary PITL formula C:

$$\models C \supset \bigcirc \Diamond C \quad \Rightarrow \quad \models \neg C \ .$$

We take C to be the implication $A^* \wedge \neg(B^*)$ and arrive at our goal $\models A^* \supset B^*$. ■

Here is two derived inference rules based on Lemma 76 which are obtainable from the axiom system:

$$\vdash_{\mathrm{PITL}} (A \wedge \mathit{more}) \supset B \quad \Rightarrow \quad \vdash_{\mathrm{PITL}} A^* \ \supset \ B^* \tag{14}$$

$$\vdash_{\mathrm{PITL}} \mathit{more} \supset (A \equiv B) \quad \Rightarrow \quad \vdash_{\mathrm{PITL}} A^* \equiv B^* \ . \tag{15}$$

The deductions required for the first of these (14) are along the lines described above in the second proof of Lemma 76 for ensuring the validity of the implication $A^* \supset B^*$ and use Lemma 74 for the PTL-based reasoning. The second derived inference rule (15) can then be readily proved from (14) together with some propositional reasoning. Incidentally, we can use (15) together with the tautology given below to deduce the important equivalence $\vdash_{\mathrm{PITL}} A^* \equiv (A \wedge \mathit{more})^*$:

$$\vdash_{\mathrm{PITL}} \ \mathit{more} \quad \supset \quad A \ \equiv \ (A \wedge \mathit{more}) \ .$$

14.4. *Full Embedding of the* FL *Axiom System in the* PITL *Axiom System*

We now look how to extend the embedding of the FL axiom system in the PITL to include the remaining Inference Rule **FInf4** concerning chop-star. However, it is first necessary to use the partial embedding of the FL axiom system to establish the deducibility of a useful PITL theorem concerning chop-star.

The derived PITL inference rule (15) in the previous Subsect. 14.3 is employed to obtain the PITL Replacement Theorem which is more powerful than the previous restricted Lemma 75. In particular, the Replacement Theorem permits the replacement of deducibly equivalent formulas even within the scope of chop-star constructs:

THEOREM 77 (REPLACEMENT THEOREM FOR PITL). — *Let B_1 and B_2 be provably equivalent formulas (i.e., $\vdash_{\mathrm{PITL}} B_1 \equiv B_2$). Suppose A_1 is an arbitrary formula and the formula A_2 is obtained from A_1 by replacing zero or more instances of B_1 in A_1 by B_2. It follows that A_1 and A_2 are provably equivalent, i.e., $\vdash_{\mathrm{PITL}} A_1 \equiv A_2$.*

PROOF. — Induction can be done on A_1's syntax with each instance of B_1 regarded as atomic. The only case differing from the proof of Lemma 75 occurs when A_1 itself is not B_1 and A_1's outermost operator is chop-star. To deal with this, we use the previously mentioned derived PITL inference rule (15). The formula A_1 has the

form C_1^* for some PITL formula C_1. The formula A_2 has the form C_2^* for some PITL formula C_2 with the associated deduction $\vdash_{\mathrm{PITL}} C_1 \equiv C_2$ true by induction. From this we readily obtain the PITL theorem $\vdash_{\mathrm{PITL}} more \supset (C_1 \equiv C_2)$. The derived inference rule (15) then yields $\vdash_{\mathrm{PITL}} C_1^* \equiv C_2^*$ which is the same as our goal $\vdash_{\mathrm{PITL}} A_1 \equiv A_2$. ■

LEMMA 78 (FULL EMBEDDING OF THE FL AXIOM SYSTEM IN THE PITL AXIOM SYSTEM). — *All* FL *axioms and inference rules can be embedded in the* PITL *axiom system.*

PROOF. — The earlier Lemma 72 deals with most of the FL axiom system. Here we only need to consider Inference Rule **FInf4**. Let us recall this inference rule:

$$\vdash_{\mathrm{FL}} \ (more \ \wedge \ \langle E \rangle empty) \ \supset \ \langle F \rangle empty \quad \Rightarrow \quad \vdash_{\mathrm{FL}} \ \langle E^* \rangle X \ \supset \ \langle F^* \rangle X \ .$$

From the assumption we can deduce the following PITL theorem using the definition of the FL-chop construct in PITL together with PITL Axiom **P6** and some propositional reasoning:

$$\vdash_{\mathrm{PITL}} \ (E \ \wedge \ more) \ \supset \ F \ .$$

From this and the derived inference rule (14) for chop-star, we obtain the PITL theorem $\vdash_{\mathrm{PITL}} E^* \supset F^*$. Some simple propositional reasoning and the PITL inference rule $\boxed{\mathrm{i}}$**Gen** then together yield $\vdash_{\mathrm{PITL}} \boxed{\mathrm{i}}(E^* \supset F^*)$. We can also readily deduce $\vdash_{\mathrm{PITL}} \Box(X \supset X)$ and make use of the following instance of PITL Axiom **P8**:

$$\vdash_{\mathrm{PITL}} \ \boxed{\mathrm{i}}(E^* \supset F^*) \wedge \Box(X \supset X) \quad \supset \quad (E^*; X) \ \supset \ (F^*; X)$$

From this with $\vdash_{\mathrm{PITL}} \boxed{\mathrm{i}}(E^* \supset F^*)$ and $\vdash_{\mathrm{PITL}} \Box(X \supset X)$ combined with propositional reasoning we obtain the PITL theorem $\vdash_{\mathrm{PITL}} (E^*; X) \supset (F^*; X)$. This can be re-expressed in FL notation as our goal $\vdash_{\mathrm{PITL}} \langle E^* \rangle X \ \supset \ \langle F^* \rangle X$. ■

15. Completeness of the PITL Axiom System

The PITL axiom system is now shown to be complete by reducing PITL formulas to ones to FL formulas. An inductive argument reduces completeness for arbitrary PITL formulas to completeness for simpler ones. The following notion of formula complexity provides a suitable hierarchy:

DEFINITION 79 (LEFT HEIGHT OF A FORMULA). — *The* left height *of a* PITL *formula A is the maximum leftward nesting of chop and chop-star constructs in A. We regard the* PTL *constructs $\bigcirc$ and $\Diamond$ as being primitives. Below is a precise definition of left height as a function denoted $lh(A)$:*

- $lh(true) = lh(P) = lh(skip) = 0$, *where P is any propositional variable.*
- $lh(\neg A) = lh(\bigcirc A) = lh(\Diamond A) = lh(A)$
- $lh(A \vee B) = \max(lh(A), lh(B)$
- $lh(A; B) = \max(lh(A) + 1, lh(B))$

– $lh(A^*) = lh(A) + 1$.

DEFINITION 80 (THE SETS PITL^0, PITL^1, ...). — *For any $n \geq 0$, PITL^n denotes the set of all* PITL *formulas having maximum left height n.*

DEFINITION 81 (EXPOSED SUBFORMULAS). — *Let A be some* PITL *formula and let B be a* PITL *subformula of A, possibly A itself. Any occurrence of B in A which is neither within the left side of any chop constructs nor within the scope of any chop-star constructs is called* exposed.

DEFINITION 82 (THE FUNCTION $d(A)$). — *For any* PITL *formula A, let $d(A)$ denote some element of* FE *which is semantically equivalent to A, i.e., $\models A \equiv d(A)$.*

LEMMA 83. — *The function d is total.*

PROOF. — Let V be the set of variables occurring in A. Lemmas 32 and 34 together ensure that a suitable element of FE_V exists with is semantically equivalent to A. Therefore d is a total function. ■

LEMMA 84. — *For any $n \geq 0$ and element A in PITL^n, the equivalence $A \equiv \langle d(A) \rangle empty$ is a* PITL *theorem.*

PROOF. — We do induction on n:

Base case for $n = 0$: It is not hard to see that every element A of PITL^0 is in fact a PTL formula. In addition, the equivalence $A \equiv d(A)$ is valid. Now it follows that the related FL formula $A \equiv \langle d(A) \rangle empty$ is also valid. The completeness of FL therefore ensures this equivalence's deducibility as a theorem in FL and hence it is also a PITL theorem by Lemma 78.

Induction step from n to $n + 1$: By induction we already have deductive equivalences for elements of PITL^n. Let A be a formula in PITL^{n+1}. Now the left-operands of A's exposed chop constructs and the main operands of A's exposed chop-star operands are all in PITL^n. Let us denote them as $B_1, \ldots, B_k$ for some $k \geq 0$. By induction, for each such formula B_i we have the PITL theorem $\vdash_{\mathrm{PITL}} B_i \equiv \langle d(B_i) \rangle empty$. Furthermore it is quite easy to deduce from this the PITL theorem $\vdash_{\mathrm{PITL}} B_i \equiv d(B_i)$ by using the deductive equivalence $\vdash_{\mathrm{PITL}} \langle d(B_i) \rangle empty \equiv d(B_i)$ which is really just an instance of Axiom **P6** in Table 1.

We then use the PITL Replacement Theorem 77 to obtain a new PITL formula A' in which each B_i is replaced by the corresponding FE formula $d(B_i)$ by using the previous deduced PITL equivalence $\vdash_{\mathrm{PITL}} B_i \equiv d(B_i)$. Consequently, the PITL theorem $\vdash_{\mathrm{PITL}} A \equiv A'$ holds. If A contains any exposed chop-stars, then A' also does and is consequently not necessarily a well-formed FL formula. However, we can once again invoke the PITL Replacement Theorem to replace every such chop-star of the form B_i^* by the deductively equivalent FL formula $\langle d(B_i)^* \rangle empty$. Each exposed $\bigcirc$ and $\Diamond$ operator in A' can be left untouched even though they are defined in terms of chop. This is because in the definition of $lh(A)$ they are treated specially and do not affect A's left height. However, their operands can influence the value of $lh(A)$ and are therefore analysed in the course of obtaining A'. Let X be the FL formula

obtained after dealing with any exposed chop-stars in A'. It is deductively equivalent to A' in the PITL axiom system and hence also to A. Hence we can deduce as a PITL theorem the equivalence $\vdash_{\mathrm{PITL}} A \equiv X$. In addition, X is a well-formed FL formula. Now the FL formula $X \equiv \langle d(A) \rangle empty$ is valid and by the completeness of FL a theorem. Lemma 78 ensures that it is also a PITL theorem. Transitivity of the deductive equivalences yields the goal $\vdash_{\mathrm{PITL}} A \equiv \langle d(A) \rangle empty$. ■

LEMMA 85 (COMPLETENESS OF PITL RELATIVE TO FL). — *Completeness holds for all* PITL *formulas relative to* FL *formulas.*

PROOF. — Let A be some PITL formula and let n be its left height. Consequently, A is in the set PITL^n. The previous Lemma 84 therefore yields the PITL theorem $\vdash_{\mathrm{PITL}} A \equiv \langle d(A) \rangle empty$. Lemma 16 then ensures that PITL completeness holds for A relative to $\langle d(A) \rangle empty$. Now $\langle d(A) \rangle empty$ itself is in FL and hence PITL completeness trivially holds for it relative to all FL formulas. By transitivity, PITL completeness holds for A relative to the set of FL formulas. ■

LEMMA 86 (COMPLETENESS OF PITL). — *The* PITL *axiom system is complete.*

PROOF. — This follows from completeness of the FL axiom system (Theorem 70), its embeddability in the PITL axiom system (Lemma 78) and Lemma 85 concerning completeness for the set of PITL formulas relative to the set of FL formulas. ■

16. Discussion

Fusion expressions and Fusion Logic could find practical application in logic specifications where sequential composition of temporal formulas, while-loops and related concepts arise (see Remarks 35) but full PITL is not needed. The FL completeness proof even suggests a decision procedure for FL based on a reduction to PTL with finite time. However, we have not yet analysed the complexity of this. It might be useful to extend fusion expressions to include an operator for temporal projection to deal with different time granularities (see [MOS 86, MOS 95]). The projection operator could at least in principle be readily added to the decision procedure by transforming formulas containing the construct into ones without it as described in [MOS 95].

Our completeness proof might generalise handle quantifier-free PITL and FL with both finite time and and past time. However, in the case of infinite time, either quantifiers or nontrivial inference rules seem unavoidable. As evidence of this, we mention Henriksen and Thiagarajan's *Dynamic Linear Time Temporal Logic* [HEN 97, HEN 99] which is similar to Fusion Logic with infinite time and requires special inference rules involving transitions and sets of words. Consequently, the axiom system is reminiscent of the one by Rosner and Pnueli [ROS 86] for quantifier-free PITL and infinite time since that also contains an inference rule involving a table of transitions.

As already noted in the introduction, our research on intervals and ITL has lead to some unexpected interesting spin-offs concerning the logics PTL and PDL. More such discoveries may be possible and we hope to pursue work in this direction.

Acknowledgements

We thank the anonymous referees for their suggestions and requests for more details which helped to improve the presentation. Part of the research described here has been kindly supported by EPSRC research grant GR/K25922.

17. References

[BAN 86] BANIEQBAL B., BARRINGER H., "A Study of an Extended Temporal Logic and a Temporal Fixed Point Calculus", report num. UMCS-86-10-2, Oct. 1986, Dept. of Computer Science, University of Manchester, England, revised June 1987.

[BEE 01] BEER I., BEN-DAVID S. et al., "The Temporal Logic Sugar", BERRY G., COMON H., FINKEL A., Eds., *13th Conference on Computer-Aided Verification (CAV01), Paris, France, 18–22 July 2001*, vol. 2102 of *LNCS*, Berlin, 2001, Springer-Verlag, p. 363–367.

[BOW 98] BOWMAN H., THOMPSON S. J., "A Tableaux Method for Interval Temporal Logic with Projection", *TABLEAUX'98, International Conference on Analytic Tableaux and Related Methods*, vol. 1397 of *Lecture Notes in AI*, Springer-Verlag, May 1998, p. 108–123.

[BOW 00] BOWMAN H., THOMPSON S. J., "A Complete Axiomatization of Interval Temporal Logic with Projection", Technical Report num. 6-00, Jan. 2000, Computing Laboratory, University of Kent, Canterbury, Great Britain.

[BOW 03] BOWMAN H., THOMPSON S. J., "A Decision Procedure and Complete Axiomatization of Finite Interval Temporal Logic with Projection", *Journal of Logic and Computation*, vol. 13, num. 2, 2003, p. 195–239, Oxford University Press.

[BÜC 62] BÜCHI J. R., "On a Decision Method in Restricted Second-Order Arithmetic", *Proc. Int. Congress on Logic, Methodology, and Philosophy of Science 1960*, Stanford, California, 1962, Stanford University Press, p. 1–12, Reprinted in [BÜC 90, pp. 425–435].

[BÜC 90] BÜCHI J. R., *The Collected Works of J. Richard Büchi*, S. Mac Lane and D. J. Siefkes, editors, Springer-Verlag, New York, 1990.

[CAU 97] CAU A., ZEDAN H., "Refining Interval Temporal Logic Specifications", BERTRAN M., RUS T., Eds., *Transformation-Based Reactive Systems Development*, vol. 1231 of *LNCS*, AMAST, Springer-Verlag, 1997, p. 79–94.

[CHA 81] CHANDRA A., HALPERN J., MEYER A., , PARIKH R., "Equations between Regular Terms and an Application to Process Logic", *Proceedings of the 13-th Annual ACM Symposium on Theory of Computing*, Milwaukee, Wisconsin, May 1981, p. 384–390.

[CHE 80] CHELLAS B. F., *Modal Logic: An Introduction*, Cambridge University Press, Cambridge, England, 1980.

[CHO 83] CHOUEKA Y., PELEG D., "A Note on ω-Regular Languages", *Bulletin of the European Association for Theoretical Computer Science*, vol. 21, 1983, p. 21–23.

[DUT 95] DUTERTRE B., "Complete Proof Systems for First Order Interval Temporal Logic", *Proc. of the 10th Annual IEEE Symposium on Logic in Computer Science*, Los Alamitos, Calif., USA, June 1995, IEEE Computer Society Press, p. 36–43.

[EME 90] EMERSON E. A., "Temporal and Modal Logic", VAN LEEUWEN J., Ed., *Handbook of Theoretical Computer Science*, vol. B: Formal Models and Semantics, chapter 16,

p. 995–1072, Elsevier/MIT Press, Amsterdam, 1990.

[FIS 77] FISCHER M. J., LADNER R. E., "Propositional Modal Logic of Programs (Extended Abstract)", *Conference Record of the Ninth Annual ACM Symposium on Theory of Computing*, Boulder, Colorado, 2–4 May 1977, p. 286–294.

[FIS 79] FISCHER M. J., LADNER R. E., "Propositional Dynamic Logic of Regular Programs", *Journal of Computer and System Sciences*, vol. 18, num. 2, 1979, p. 194–211.

[FRE 03] FRENCH T., REYNOLDS M., "A Sound and Complete Proof System for QPTL", BALBIANI P., SUZUKI N.-Y., WOLTER F., ZAKHARYASCHEV M., Eds., *Advances in Modal Logic*, vol. 4, p. 127–148, King's College Publications, London, 2003.

[GAB 80] GABBAY D., PNUELI A., SHELAH S., STAVI J., "On the Temporal Analysis of Fairness", *Seventh Annual ACM Symposium on Principles of Programming Languages*, 1980, p. 163–173.

[HAL 83] HALPERN J., MANNA Z., MOSZKOWSKI B., "A Hardware Semantics Based on Temporal Intervals", DIAZ J., Ed., *Proceedings of the 10-th International Colloquium on Automata, Languages and Programming*, vol. 154 of *LNCS*, Berlin, 1983, Springer-Verlag, p. 278–291.

[HAM 92] HAMAGUCHI K., HIRAISHI H., YAJIMA S., "Infinity-Regular Temporal Logic and Its Model Checking Problem", *Theor. Comp. Sci.*, vol. 103, num. 2, 1992, p. 191–204.

[HAR 84] HAREL D., "Dynamic Logic", GABBAY D., GUENTHNER F., Eds., *Handbook of Philosophical Logic*, vol. II, p. 497–604, Reidel Publishing Company, Dordrecht, 1984.

[HAR 00] HAREL D., KOZEN D., TIURYN J., *Dynamic Logic*, MIT Press, Cambridge, Massachusetts, 2000.

[HAR 02] HAREL D., KOZEN D., TIURYN J., "Dynamic Logic", GABBAY D., GUENTHNER F., Eds., *Handbook of Philosophical Logic*, vol. 4, p. 99–217, Kluwer Academic Publishers, Dordrecht, 2nd edition edition, 2002.

[HEN 97] HENRIKSEN J. G., THIAGARAJAN P. S., "Dynamic Linear Time Temporal Logic", report num. RS-97-8, April 1997, BRICS, Department of Computer Science, University of Aarhus, Aarhus, Denmark, Available at `http://www.brics.dk/RS/97/8/`.

[HEN 99] HENRIKSEN J. G., THIAGARAJAN P. S., "Dynamic Linear Time Temporal Logic", *Annals of Pure and Applied Logic*, vol. 96, num. 1-3, 1999, p. 187–207.

[HIR 92] HIRAISHI H., HAMAGUCHI K., FUJII H., YAJIMA S., "Regular Temporal Logic Expressively Equivalent to Finite Automata and its Application to Logic Design Verification", *Journal of Information Processing*, vol. 15, num. 1, 1992, p. 129–138.

[HOP 79] HOPCROFT J. E., ULLMAN J. D., *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley, Reading, Mass., 1979.

[HUG 96] HUGHES G. E., CRESSWELL M. J., *A New Introduction to Modal Logic*, Routledge, London, 1996.

[JON 83] JONES C. B., "Specification and Design of (Parallel) Programs", MASON R. E. A., Ed., *Proceedings of IFIP Congress '83*, Amsterdam, 1983, North Holland Publishing Co., p. 321–332.

[KES 95] KESTEN Y., PNUELI A., "A Complete Proof System for QPTL", *Proc. 10th IEEE Symp. on Logic in Computer Science*, IEEE Computer Society Press, 1995, p. 2–12.

[KES 02] KESTEN Y., PNUELI A., "Complete Proof System for QPTL", *Journal of Logic and Computation*, vol. 12, num. 5, 2002, p. 701–745.

[KON 95] KONO S., "A Combination of Clausal and Non-Clausal Temporal Logic Programs", FISHER M., OWENS R., Eds., *Executable Modal and Temporal Logics*, vol. 897 of *LNCS*, Berlin, Feb. 1995, Springer-Verlag, p. 40–57.

[KOZ 90] KOZEN D., TIURYN J., "Logics of Programs", VAN LEEUWEN J., Ed., *Handbook of Theoretical Computer Science*, vol. B, p. 789–840, Elsevier Science Publishers, Amsterdam, 1990.

[KOZ 94] KOZEN D., "A Completeness Theorem for Kleene Algebras and the Algebra of Regular Events", *Inf. and Comp.*, vol. 110, num. 2, 1994, p. 366–390.

[LIC 85] LICHTENSTEIN O., PNUELI A., ZUCK L., "The Glory of the Past", PARIKH R. et al., Eds., *Logics of Programs*, vol. 193 of *LNCS*, Berlin, 1985, Springer-Verlag, p. 196–218.

[MOS 83a] MOSZKOWSKI B., "Reasoning about Digital Circuits", PhD thesis, Department of Computer Science, Stanford University, 1983, Technical report STAN–CS–83–970.

[MOS 83b] MOSZKOWSKI B., "A Temporal Logic for Multi-Level Reasoning about Hardware", *Proceedings of the 6-th International Symposium on Computer Hardware Description Languages*, Pittsburgh, Pennsylvania, May 1983, North-Holland Pub. Co., p. 79–90.

[MOS 85] MOSZKOWSKI B., "A Temporal Logic for Multilevel Reasoning about Hardware", *Computer*, vol. 18, 1985, p. 10–19, IEEE Computer Society Press.

[MOS 86] MOSZKOWSKI B., *Executing Temporal Logic Programs*, Cambridge University Press, Cambridge, England, 1986.

[MOS 94] MOSZKOWSKI B., "Some Very Compositional Temporal Properties", OLDEROG E.-R., Ed., *Programming Concepts, Methods and Calculi*, vol. A-56 of *IFIP Transactions*, IFIP, Elsevier Science B.V. (North–Holland), 1994, p. 307–326.

[MOS 95] MOSZKOWSKI B., "Compositional Reasoning about Projected and Infinite Time", *Proceedings of the First IEEE Int'l Conf. on Engineering of Complex Computer Systems (ICECCS'95)*, IEEE Computer Society Press, 1995, p. 238–245.

[MOS 98] MOSZKOWSKI B., "Compositional Reasoning Using Interval Temporal Logic and Tempura", DE ROEVER W.-P., LANGMAACK H., PNUELI A., Eds., *Compositionality: The Significant Difference*, vol. 1536 of *LNCS*, Berlin, 1998, Springer-Verlag, p. 439–464.

[MOS 00a] MOSZKOWSKI B., "An Automata-Theoretic Completeness Proof for Interval Temporal Logic (Extended Abstract)", MONTANARI U., ROLIM J., WELZL E., Eds., *Proceedings of the 27th International Colloquium on Automata, Languages and Programming (ICALP 2000)*, vol. 1853 of *LNCS*, Geneva, Switzerland, Jul. 2000, Springer-Verlag, p. 223–234.

[MOS 00b] MOSZKOWSKI B., "A Complete Axiomatization of Interval Temporal Logic with Infinite Time (Extended Abstract)", *Proc. of the 15th Annual IEEE Symposium on Logic in Computer Science (LICS 2000)*, IEEE Computer Society Press, June 2000, p. 242–251.

[MOS 04a] MOSZKOWSKI B., "A Hierarchical Completeness Proof for Propositional Temporal Logic", DERSHOWITZ N., Ed., *Verification: Theory and Practice: Essays Dedicated to Zohar Manna on the Occasion of His 64th Birthday*, vol. 2772 of *LNCS*, p. 480–523, Springer-Verlag, Heidelberg, 2004.

[MOS 04b] MOSZKOWSKI B., "Small Models for Propositional Dynamic Logic without Fischer-Ladner Closures", Submitted for publication, March 2004.

[NIS 79] NISHIMURA H., "Sequential Method in Propositional Dynamic Logic", *Acta Informatica*, vol. 12, 1979, p. 377–400.

[PAE 88] PAECH B., "Gentzen-Systems for Propositional Temporal Logics", BÖRGER E., BÜNING H. K., RICHTER M. M., Eds., *Proceedings of the 2nd Workshop on Computer Science Logic, Duisburg (FRG)*, vol. 385 of *LNCS*, Springer-Verlag, Oct. 1988, p. 240-253.

[PAR 85] PARIKH R., CHANDRA A. K., HALPERN J. Y., MEYER A. R., "Equations Between Regular Terms and an Application to Process Logic", *SIAM Journal on Computing*, vol. 14, num. 4, 1985, p. 935–942.

[PNU 77] PNUELI A., "The Temporal Logic of Programs", *Proceedings of the 18th Symposium on the Foundation of Computer Science*, ACM, 1977, p. 46–57.

[PRA 79] PRATT V. R., "Process Logic", *Sixth Annual ACM Symposium on Principles of Programming Languages*, 1979, p. 93–100.

[ROS 86] ROSNER R., PNUELI A., "A Choppy Logic", *First Annual IEEE Symposium on Logic in Computer Science*, IEEE Computer Society Press, June 1986, p. 306–313.

[SAL 66] SALOMAA A., "Two Complete Axiom Systems for the Algebra of Regular Events", *Journal of the ACM*, vol. 13, num. 1, 1966, p. 158–169.

[SIE 70] SIEFKES D., *Decidable Theories I: Büchi's Monadic Second Order Successor Arithmetic*, vol. 120 of *Lecture Notes in Mathematics*, Springer-Verlag, Berlin, 1970.

[SIS 87] SISTLA A. P., VARDI M. Y., WOLPER P., "The Complementation Problem for Büchi Automata with Applications to Temporal Logic", *Theor. Comp. Sci.*, vol. 49, 1987, p. 217–237.

[STO 74] STOCKMEYER L. J., "The Complexity of Decision Problems in Automata Theory and Logic", PhD thesis, MIT, Jul. 1974, Available as Project MAC Technical Report 133.

[THO 79] THOMAS W., "Star-Free Regular Sets of ω-Sequences", *Inf. and Control*, vol. 42, num. 2, 1979, p. 148–156.

[VAL 79] VALIEV M. K., "On Axiomatization of Deterministic Propositional Dynamic Logic", BECVÁR J., Ed., *Proc. of Mathematical Foundations of Computer Science (MFCS) 1979, 8th Symp.*, vol. 74 of *LNCS*, Springer-Verlag, 1979, p. 482–491.

[WAG 76] WAGNER K., "Eine Axiomatisierung der Theorie der Regulären Folgenmengen", *Elektronische Informationsverarbeitung und Kybernetik EIK*, vol. 12, num. 7, 1976, p. 337–354.

[Wan 99] WANG HANPIN, XU QIWEN, "Temporal Logics over Infinite Intervals", report num. 158, 1999, UNU/IIST, Macau.

[WOL 82] WOLPER P. L., "Specification and Synthesis of Communicating Processes Using an Extended Temporal Logic", PhD thesis, Department of Computer Science, Stanford University, 1982.

[WOL 83] WOLPER P. L., "Temporal Logic Can be More Expressive", *Information and Control*, vol. 56, num. 1-2, 1983, p. 72–99.

[Zho 91] ZHOU CHAOCHEN, HOARE C. A. R., RAVN A. P., "A Calculus of Durations", *Information Processing Letters*, vol. 40, num. 5, 1991, p. 269–276.

[Zho 04] ZHOU CHAOCHEN, HANSEN M. R., *Duration Calculus: A Formal Approach to Real-Time Systems*, Monographs in Theoretical Computer Science (An EATCS series), Springer-Verlag, 2004.

A. The Complexity of PITL

We briefly look at the complexity of PITL. This material is adapted from our doctoral dissertation [MOS 83a, pages 20–24] and has not previously appeared in print elsewhere. It was done in fruitful collaboration with Joseph Y. Halpern.

A.1. *Undecidability of* PITL *with Interval Variables*

PITL can be defined so that propositional variables are interval-based. Instead of each state in an interval mapping variables to values, every subinterval has its own mapping.

THEOREM 87 (HALPERN AND MOSZKOWSKI). — *Satisfiability for* PITL *with interval variables is undecidable.*

PROOF. — Our proof is very similar to the one presented by Chandra et al. [CHA 81, PAR 85] for showing the undecidability of satisfiability for a propositional process logic. We strengthen their result since we do not require programs in order to obtain undecidability.

Given two context-free grammars G_1 and G_2, we can construct a PITL formula that is satisfiable iff the intersection of the languages generated by G_1 and G_2 is nonempty. Since this intersection problem is undecidable [HOP 79], it follows that satisfiability for PITL is also.

Without lose of generality, we assume that G_1 and G_2 contain no ϵ-productions, use 0 and 1 as the only terminal symbols and are in Greibach normal form (that is, the right-hand side of each production starts with a terminal symbol).

For a given an interval $s_0 \dots s_n$ and an interpretation $\mathcal{M}$, we form the *trace* $\tau_{s_0 \dots s_n}(P)$ of a variable P by observing P's behaviour over the states $s_0, \dots, s_n$. We define τ as follows:

$$\begin{aligned} \tau_s(P) &= \begin{cases} 0 & \text{if } \mathcal{M}_s[\![P]\!] = \mathit{false} \\ 1 & \text{if } \mathcal{M}_s[\![P]\!] = \mathit{true} \end{cases} \\ \tau_{s_0 \dots s_n}(P) &= \tau_{s_0}(P) \dots \tau_{s_n}(P) \ . \end{aligned}$$

Suppose that G is a context-free grammar consisting of a list π of m production sets $\pi_1, \dots, \pi_m$, one for each nonterminal symbol N_i:

$$\begin{aligned} \pi_1 &: \ N_1 \to \pi_{11} \mid \pi_{12} \mid \dots \mid \pi_{1,|\pi_1|} \\ \pi_2 &: \ N_2 \to \pi_{21} \mid \pi_{22} \mid \dots \mid \pi_{2,|\pi_2|} \\ &\vdots \\ \pi_m &: \ N_m \to \pi_{m1} \mid \pi_{m2} \mid \dots \mid \pi_{m,|\pi_m|} \ . \end{aligned}$$

Let $L(G, N_i)$ be the language generated by G with N_i as the start symbol. We give a translation $f(G, N_i)$ into PITL such that an interval $s_0 \ldots s_n$ satisfies $f(G, N_i)$ iff P's trace in $s_0 \ldots s_n$ is in $L(G, N_i)$:

$$s_0 \ldots s_n \models f(G, N_i) \quad \text{iff} \quad \tau_{s_0 \ldots s_n}(P) \in L(G, N_i) \ . \tag{16}$$

For each of the production sets π_i, the associated translation $f(\pi_i)$ is the PITL formula

$$\boxed{a}\big(N_i \equiv (f(\pi_{i1}) \vee f(\pi_{i2}) \vee \cdots \vee f(\pi_{i,|\pi_i|}))\big) \ .$$

Each production string $\pi_{ij} = V_1 V_2 \ldots V_{|\pi_{ij}|}$ has the translation

$$f(V_1 V_2 \ldots V_m) = f(V_1); \mathit{skip}; f(V_2); \mathit{skip}; \ldots \mathit{skip}; f(V_{|\pi_{ij}|})$$

where

$$\begin{array}{lcl} f(0) & = & (\neg P \wedge \mathit{empty}) \\ f(1) & = & (P \wedge \mathit{empty}) \\ f(N_i) & = & N_i, \quad \text{for each nonterminal symbol } N_i \ . \end{array}$$

Recall that the variable P determines whether a state maps to 0 or 1. In order to avoid conflicts, we require that P not occur in the grammar. The overall translation $f(G, N_i)$ is as follows:

$$N_i \wedge f(\pi) \ .$$

It is now easy to show (16) by induction on the size of the interval $s_0 \ldots s_n$. We need the grammar to be in Greibach normal form in order for the inductive step to go through. See Chandra et al. [CHA 81, PAR 85] for details.

Given two context-free grammars G_1 and G_2 with disjoint sets of nonterminals and respective start symbols S_1 and S_2, the PITL formula

$$f(G_1, S_1) \wedge f(G_2, S_2)$$

is satisfiable iff the intersection of the languages $L(G_1)$ and $L(G_2)$ is nonempty. Because this emptiness problem is undecidable [HOP 79], it follows that satisfiability in PITL is also. ■

COROLLARY 88. — *Validity for* PITL *with interval variables is undecidable.*

REMARKS 89. — Undecidability can be shown to hold even if we are restricted to just using *empty* instead of *skip*. To do this, we use interval-based propositional variables P and Q. The following operators *beg* and *fin* for testing at the beginning and end of intervals are used:

$$\mathit{beg}\ A \quad \stackrel{\text{def}}{\equiv} \quad \boxed{i}(\mathit{empty} \supset A) \tag{17}$$

$$\mathit{fin}\ A \quad \stackrel{\text{def}}{\equiv} \quad \Box(\mathit{empty} \supset A) \ . \tag{18}$$

We introduce an operator $\mathit{group}(P, Q)$ which is true in intervals satisfying the next formula:

$$(\boxed{a}\ \mathit{beg}\ Q);\ \mathit{skip};\ (\boxed{a}\ \mathit{beg}\ (P \wedge \neg Q));\ \mathit{skip};\ (\boxed{a}\ \mathit{beg}\ Q) \ .$$

Such intervals are in effect delimited on both sides by states with Q true and contain internal states with $P \wedge \neg Q$ true. Hence, Q acts as a delimiter around a group of states where P is true. The following is a sample 5-state interval $s_0 \ldots s_4$ satisfying $group(P, Q)$:

$$\begin{array}{ccccc} s_0 & s_1 & s_2 & s_3 & s_4 \\ Q & P & P & Q & Q \\ & \wedge & \wedge & & \\ & \neg Q & \neg Q & & \end{array} \ .$$

Similarly, $group(\neg P, Q)$ denotes a delimited group of states with $\neg P$ true in the interior. If we take $empty$ as a primitive operator, the operator $group$ can be expressed without the use of the operator $\bigcirc$:

$$group(P, Q) \quad \stackrel{\text{def}}{\equiv} \quad grp(P, Q) \wedge \neg(grp(P, Q); grp(P, Q)) \ ,$$

where $grp(P, Q)$ has the definition below:

$$grp(P, Q) \quad \stackrel{\text{def}}{\equiv} \quad beg\ Q \wedge fin\ Q \wedge \boxed{a}(beg\ (P \wedge \neg Q) \vee Q) \wedge \diamondsuit_a\ beg\ P \ .$$

The modified translation f' is like f with the following exceptions:

$$\begin{aligned} f'(V_1 V_2 \ldots V_m) &= f'(V_1); f'(V_2); \ldots; f'(V_m) \\ f'(0) &= group(\neg P, Q) \\ f'(1) &= group(P, Q) \ . \end{aligned}$$

END OF REMARKS 89.

A.2. *Decidability of Local* PITL

We normally restrict propositional variables to be state-based. The associated standard version of PITL is often called *Local PITL*.

THEOREM 90 (HALPERN AND MOSZKOWSKI). — *Satisfiability for local* PITL *with quantification is decidable*[3].

PROOF. — We give a linear translation from formulas in PITL to formulas into *Quantified Propositional Temporal Logic* (QPTL). Formulas are built from propositional variables P, Q, ... and the following constructs:

$$\neg X \quad Y \wedge Z \quad \bigcirc X \quad \Box X \quad \exists P.\, X \ ,$$

where X, Y and Z are themselves QPTL formulas. The interpretation of variables and formulas is identical to that of local PITL with quantification. The particular QPTL used by us restricts intervals to be finite and is known as *Weak* QPTL

3. The definition of PITL in our dissertation did not include chop-star. However, the proof can be extended to handle it and we leave the details as an exercise.

(WQPTL). It can express such constructs as $\Diamond X$, $Y\ until\ Z$ (strong until) and $empty$. Wolper [WOL 82, SIS 87] shows that the theory of QPTL over infinite intervals is decidable but nonelementary; this result easily extends to WQPTL (see also [LIC 85] for a later direct proof). The complexity is elementary in the alternation of $\neg$ and $\exists$.

For a given variable P and PITL formula A, we now give a translation $g(P, A)$ which is true of an interval $s_0 \ldots s_n$ in weak QPTL iff the variable P is true for the first time in some state s_i and A is true over the initial interval $s_0 \ldots s_i$. Thus, $g(P, A)$ is semantically like the PITL formula shown below:

$$\Diamond\!\!\!\!\cdot(\Box(P \equiv empty) \wedge A) \ .$$

This specifies an interval containing a prefix subinterval which terminates exactly when P becomes true and also satisfies A. The subformula $\Box(P \equiv empty)$ can be abbreviated as $halt\ P$.

Here is the definition of g (see the previous footnote 3):

$$\begin{array}{lcl}
g(P, Q) & = & (\Diamond P) \wedge Q \\
g(P, \neg A) & = & \neg g(P, A) \wedge \Diamond P \\
g(P, (A \wedge B)) & = & g(P, A) \wedge g(P, B) \\
g(P, \bigcirc A) & = & \neg P \wedge \bigcirc g(P, A) \\
g(P, (A; B)) & = & \exists R.\, \big[g(R, A) \wedge \big((\neg P)\ until\ (R \wedge g(P, B))\big)\big], \\
 & & \quad \text{where } R \text{ does not occur freely in either } A \text{ or } B. \\
g(P, \exists Q.\, A) & = & \exists Q.\, g(P, A) \ .
\end{array}$$

A formula A in PITL has the same semantics as $g(empty, A)$ in WQPTL:

$$s_0 \ldots s_n \models A \quad \text{iff} \quad s_0 \ldots s_n \models_{\mathrm{WQPTL}} g(empty, A) \ .$$

■

REMARKS 91. — The translation can be extended to handle PITL over infinite intervals.

A.3. *Lower Bound for Satisfiability*

The complexity of decidability for PITL is connected to that testing generalised regular expressions containing a complement operator. D. Kozen (private communication) proved the following theorem:

THEOREM 92 (D. KOZEN). — *Satisfiability for local* PITL *is nonelementary.*

PROOF. — Stockmeyer [STO 74] shows that the problem of deciding the emptiness of an arbitrary regular expression over the alphabet $\{0, 1\}$ and with operators $\cup$ (union), $\cdot$ (concatenation) and $\sim$ (complement) is nonelementary. Satisfiability for local PITL is now reduced to this problem. Given a regular expression e, we construct

a PITL formula $h(e)$ containing instances of a single propositional variable P and which is satisfiable iff the language generated by e is nonempty. The definition of h given by induction on the syntactic structure of e:

e	$h(e)$
0	$\neg P \wedge \mathit{empty}$
1	$P \wedge \mathit{empty}$
$e_1 \cup e_2$	$h(e_1) \vee h(e_2)$
$\sim e$	$\neg h(e)$
$e_1 \cdot e_2$	$h(e_1); \mathit{skip}; h(e_2)$

For example, the translation of the regular expression $(01) \cup \sim 1$ is as follows:

$$\big((\neg P \wedge \mathit{empty}); \mathit{skip}; (P \wedge \mathit{empty})\big) \vee \neg(P \wedge \mathit{empty}) \ .$$

Note that the length of $h(e)$ is linear in that of e.

A formal proof relating nonemptiness of a regular expression e and satisfiability of the PITL formula $h(e)$ would use a straightforward induction on the syntactic structure of e. ■

B. Deduction of PTL Axioms from the FL Axiom System

This appendix contains various FL theorems and their deductions. These include ones corresponding to some of the PTL axioms in Table 3 in Section 10. Most of the PTL axioms and inference rules have identical or nearly identical versions in the FL axiom system in Table 2 in Section 9. The three exceptions are Axioms **A1**, **A3** and **A4**. We will look at each of them in turn as FL theorems **T5**, **T7** and **T13**, respectively. The trickiest is Axiom **A1**. The symbol $\vdash$ as used here always refers to $\vdash_{\mathrm{FL}}$. None of the FE formulas occurring in the proofs contain variables and therefore the proofs also ensure well-formed FL_V theorems and derived inference rules for any V.

T3 $\quad \vdash \quad \Box(X \supset Y) \quad \supset \quad \Diamond X \supset \Diamond Y$

Proof:

1	$\vdash$	$\Box(X \supset Y) \quad \supset \quad \langle \mathit{skip}^* \rangle X \supset \langle \mathit{skip}^* \rangle Y$	**FL9**
2	$\vdash$	$\Diamond X \quad \equiv \quad \langle \mathit{skip}^* \rangle X$	**FL3**
3	$\vdash$	$\Diamond Y \quad \equiv \quad \langle \mathit{skip}^* \rangle Y$	**FL3**
4	$\vdash$	$\Box(X \supset Y) \quad \supset \quad \Diamond X \supset \Diamond Y$	2,3,**Prop**

The following slightly obscure theorem is used in the proof of **T5**:

T4 $\quad \vdash \quad \Box(\neg Y \supset \neg X) \quad \supset \quad \Box X \supset \Box Y$

Proof:

1	$\vdash$	$\Box(\neg Y \supset \neg X) \supset \Diamond\neg Y \supset \Diamond\neg X$	**T3**
2	$\vdash$	$\Box(\neg Y \supset \neg X) \supset \neg\Diamond\neg X \supset \neg\Diamond\neg Y$	1,**Prop**
3	$\vdash$	$\Box(\neg Y \supset \neg X) \supset \Box X \supset \Box Y$	2,def. of $\Box$

Below is the proof of PTL Axiom **A1** as FL theorem **T5**. In the final step, $\supset$**-chain** stands for a chain of implications.

T5 $\vdash \Box(X \supset Y) \supset \Box X \supset \Box Y$

Proof:

1	$\vdash$	$(X \supset Y) \supset (\neg Y \supset \neg X)$	**Prop**
2	$\vdash$	$\neg(\neg Y \supset \neg X) \supset \neg(X \supset Y)$	1,**Prop**
3	$\vdash$	$\Box\big(\neg(\neg Y \supset \neg X) \supset \neg(X \supset Y)\big)$	2,$\Box$**Gen**
4	$\vdash$	$\Box\big(\neg(\neg Y \supset \neg X) \supset \neg(X \supset Y)\big)$ $\supset \Box(X \supset Y) \supset \Box(\neg Y \supset \neg X)$	**T4**
5	$\vdash$	$\Box(X \supset Y) \supset \Box(\neg Y \supset \neg X)$	3,4,**MP**
6	$\vdash$	$\Box(\neg Y \supset \neg X) \supset \Box X \supset \Box Y$	**T4**
7	$\vdash$	$\Box(X \supset Y) \supset \Box X \supset \Box Y$	5,6,$\supset$**-chain**

T6 $\vdash \bigcirc\neg X \supset \neg\bigcirc X$

Proof:

1	$\vdash$	$\bigcirc X \supset \circledcirc X$	**FL10**
2	$\vdash$	$\bigcirc X \supset \neg\bigcirc\neg X$	1,def. of $\circledcirc$
3	$\vdash$	$\bigcirc\neg X \supset \neg\bigcirc X$	2,**Prop**

Here is a proof of PTL Axiom **A3**:

T7 $\vdash \bigcirc(X \supset Y) \supset \bigcirc X \supset \bigcirc Y$

Proof:

1	$\vdash$	$\langle skip\rangle(\neg X \vee Y) \supset (\langle skip\rangle\neg X) \vee (\langle skip\rangle Y)$	**FL7**
2	$\vdash$	$\bigcirc(\neg X \vee Y) \equiv \langle skip\rangle(\neg X \vee Y)$	**FL2**
3	$\vdash$	$\bigcirc\neg X \equiv \langle skip\rangle\neg X$	**FL2**
4	$\vdash$	$\bigcirc Y \equiv \langle skip\rangle Y$	**FL2**
5	$\vdash$	$\bigcirc(\neg X \vee Y) \supset \bigcirc\neg X \vee \bigcirc Y$	1–4,**Prop**
6	$\vdash$	$\bigcirc\neg X \supset \neg\bigcirc X$	**T6**
7	$\vdash$	$\bigcirc(\neg X \vee Y) \supset \neg\bigcirc X \vee \bigcirc Y$	5,6,**Prop**
8	$\vdash$	$\bigcirc(X \supset Y) \supset \bigcirc X \supset \bigcirc Y$	7,def. of $\supset$

The remaining proofs are for ultimately deducing PTL Axiom **A4** as FL theorem **T13**. The following derived rule **DR1** can be readily generalised to allow some arbitrary FE formula in place of $skip$. In addition, a version can be proven which uses $\equiv$ instead of $\supset$.

DR1 $\vdash X \supset Y \Rightarrow \vdash \langle skip\rangle X \supset \langle skip\rangle Y$

Proof:

1	$\vdash \quad X \supset Y$	assump.
2	$\vdash \quad \Box(X \supset Y)$	$\Box$**Gen**
3	$\vdash \quad \Box(X \supset Y) \quad \supset \quad \langle skip\rangle X \supset \langle skip\rangle Y$	**FL9**
4	$\vdash \quad \langle skip\rangle X \quad \supset \quad \langle skip\rangle Y$	2,3,**MP**

DR2 $\quad \vdash X \supset Y \quad \Rightarrow \quad \bigcirc X \supset \bigcirc Y$

Proof:

1	$\vdash \quad X \quad \supset \quad Y$	assump.
2	$\vdash \quad \langle skip\rangle X \quad \supset \quad \langle skip\rangle Y$	1,**DR1**
3	$\vdash \quad \bigcirc X \quad \equiv \quad \langle skip\rangle X$	**FL2**
4	$\vdash \quad \bigcirc Y \quad \equiv \quad \langle skip\rangle Y$	**FL2**
5	$\vdash \quad \bigcirc X \quad \supset \quad \bigcirc Y$	3,4,**Prop**

DR3 $\quad \vdash X \equiv Y \quad \Rightarrow \quad \bigcirc X \equiv \bigcirc Y$

Proof:

1	$\vdash \quad X \quad \equiv \quad Y$	assump.
2	$\vdash \quad X \quad \supset \quad Y$	1,**Prop**
3	$\vdash \quad \bigcirc X \quad \supset \quad \bigcirc Y$	2,**DR2**
4	$\vdash \quad Y \quad \supset \quad X$	1,**Prop**
5	$\vdash \quad \bigcirc Y \quad \supset \quad \bigcirc X$	4,**DR2**
6	$\vdash \quad \bigcirc X \quad \equiv \quad \bigcirc Y$	3,5,**Prop**

T8 $\quad \vdash \quad \Diamond X \quad \equiv \quad X \vee \bigcirc \Diamond X$

Proof:

1	$\vdash \quad \Diamond X \quad \equiv \quad \langle skip^* \rangle X$	**FL3**
2	$\vdash \quad \langle skip^* \rangle X \quad \equiv \quad X \vee \langle skip; skip^* \rangle X$	**FL8**
3	$\vdash \quad \langle skip; skip^* \rangle X \quad \equiv \quad \langle skip \rangle \langle skip^* \rangle X$	**FL6**
4	$\vdash \quad \bigcirc \langle skip^* \rangle X \quad \equiv \quad \langle skip \rangle \langle skip^* \rangle X$	**FL2**
5	$\vdash \quad \bigcirc \Diamond X \quad \equiv \quad \bigcirc \langle skip^* \rangle X$	1,**DR3**
6	$\vdash \quad \Diamond X \quad \equiv \quad X \vee \bigcirc \Diamond X$	1–5,**Prop**

T9 $\quad \vdash \quad X \quad \supset \quad \Diamond X$

Proof:

1	$\vdash \quad \Diamond X \quad \equiv \quad X \vee \bigcirc \Diamond X$	**T8**
2	$\vdash \quad X \quad \supset \quad \Diamond X$	1,**Prop**

T10 $\vdash \quad \bigcirc \Diamond X \quad \supset \quad \Diamond X$

Proof:

1	$\vdash \quad \Diamond X \quad \equiv \quad X \vee \bigcirc \Diamond X$	**T8**
2	$\vdash \quad \bigcirc \Diamond X \quad \supset \quad \Diamond X$	1,**Prop**

T11 $\vdash \quad \Box X \quad \supset \quad X$

Proof:

1	$\vdash \quad \neg X \quad \supset \quad \Diamond \neg X$	**T9**
2	$\vdash \quad \neg \Diamond \neg X \quad \supset \quad X$	1,**Prop**
3	$\vdash \quad \Box X \quad \supset \quad X$	2,def. of $\Box$

T12 $\vdash \quad \Box X \quad \supset \quad \textcircled{w} \Box X$

Proof:

1	$\vdash \quad \neg\neg \Diamond \neg X \quad \supset \quad \Diamond \neg X$	**Prop**
2	$\vdash \quad \bigcirc \neg\neg \Diamond \neg X \quad \supset \quad \bigcirc \Diamond \neg X$	1,**DR2**
3	$\vdash \quad \bigcirc \Diamond \neg X \quad \supset \quad \Diamond \neg X$	**T10**
4	$\vdash \quad \bigcirc \neg\neg \Diamond \neg X \quad \supset \quad \Diamond \neg X$	2,3,$\supset$**-chain**
5	$\vdash \quad \bigcirc \neg \Box X \quad \supset \quad \Diamond \neg X$	4,def. of $\Box$
6	$\vdash \quad \neg \Diamond \neg X \quad \supset \quad \neg \bigcirc \neg \Box X$	5,**Prop**
7	$\vdash \quad \Box X \quad \supset \quad \textcircled{w} \Box X$	6,def. of $\Box, \textcircled{w}$

Below is a proof of PTL Axiom **A4**:

T13 $\vdash \quad \Box X \quad \supset \quad X \wedge \textcircled{w} \Box X$

Proof:

1	$\vdash \quad \Box X \quad \supset \quad X$	**T11**
2	$\vdash \quad \Box X \quad \supset \quad \textcircled{w} \Box X$	**T12**
3	$\vdash \quad \Box X \quad \supset \quad X \wedge \textcircled{w} \Box X$	1,2,**Prop**

This concludes the proofs.