An earlier version of this appeared in *We Will Show Them: Essays in Honour of Dov Gabbay on his 60th Birthday*, Volume 2. S. Artemov, H. Barringer, A. S. d'Avila Garcez, L. C. Lamb, and J. Woods (eds.), pages 371–440, College Publications (formerly KCL Publications), King's College, London, 2005, URL: `http://www.collegepublications.co.uk`.

A Hierarchical Analysis of Propositional Temporal Logic Based on Intervals

Ben Moszkowski*
Software Technology Research Laboratory
Gateway House
De Montfort University
The Gateway
Leicester LE1 9BH
Great Britain
email: `x@y, where x=benm and y=dmu.ac.uk`

26 January 2006

Abstract

We present a hierarchical framework for analysing propositional linear-time temporal logic (PTL) to obtain standard results such as a small model property, decision procedures and axiomatic completeness. Both finite time and infinite time are considered and one consequent benefit of the framework is the ability to systematically reduce infinite-time reasoning to finite-time reasoning. The treatment of PTL with both the operator *until* and past time naturally reduces to that for PTL without either one. Our method utilises a low-level normal form for PTL called a *transition configuration*. In addition, we employ reasoning about intervals of time. Besides being hierarchical and interval-based, the approach differs from other analyses of PTL typically based on sets of formulas and sequences of such sets. Instead we describe models using time intervals represented as finite and infinite sequences of states. The analysis relates larger intervals with smaller ones. Steps involved are expressed in Propositional Interval Temporal Logic (PITL) which is better suited than PTL for sequentially combining and decomposing formulas. Consequently, we can articulate issues in PTL model construction of equal relevance in more conventional analyses but normally only considered at the metalevel. We also describe a decision procedure based on Binary Decision Diagrams.

Beyond the specific issues involving PTL, the research is a significant application of ITL and interval-based reasoning and illustrates a general approach to formally reasoning about sequential and parallel behaviour in discrete linear time. The work also includes some interesting representation theorems. In addition, it has relevance to hardware description and verification since the specification languages PSL/Sugar (now IEEE standard 1850) and 'temporal e' (part of IEEE candidate standard 1647) both contain temporal constructs concerning intervals of time as does the related SystemVerilog Assertion language contained in SystemVerilog, an extension of the IEEE 1364-2001 Verilog language.

Keywords: temporal logic, interval temporal logic, small models, decision procedures, axiomatic completeness

1 Introduction

Following the seminal paper by Pnueli [61], temporal logic [20, 40, 44] has become one of the main formalisms used in computer science for reasoning about the dynamic behaviour of systems. In particular, propositional linear-time temporal logic (PTL) and some variants of it have been extensively studied and used. In a relatively recent and significant article, Lichtenstein and Pnueli [43] give a detailed analysis of PTL which is

*Part of the research described here has been kindly supported by EPSRC research grant GR/K25922.

meant to largely subsume and supercede earlier ones. Indeed, the work appears to have the rather ambitious goal of coming close to offering the last word on the subject and is perhaps best described in the authors' own words:

> The paper summarizes work of over 20 years and is intended to provide a definitive reference to the version of propositional temporal logic used for the specification and verification of reactive systems.

The version of PTL considered by Lichtenstein and Pnueli has discrete time and past time. Both a decision procedure and axiomatic completeness are investigated and a new simplified axiom system is presented. The approach makes use of semantic tableaux and throughout the presentation the treatment of PTL with past-time operators runs in parallel with the future-only version. The authors choose in particular to use tableaux since they offer a basis for uniformly showing axiomatic completeness and also obtaining a practical decision procedure. The extensive material about past time is distinctly marked so that one can optionally delete it to obtain an analysis limited to the future fragment of PTL.

We present a novel framework for investigating PTL which significantly differs from the methods of Lichtenstein and Pnueli and earlier treatments such as [27, 28, 40, 73]. It is used to obtain standard results such as a small model property, a practical decision procedure and axiomatic completeness. However, instead of relying on semantic tableaux, filtration and other previous techniques, our method is based on an interval-oriented analysis of certain kinds of low-level PTL formulas called *transition configurations*. An important feature of this approach is that it provides a natural hierarchical means of reducing full PTL to this subset and also reduces both PTL with the *until* operator and past time to versions without them. Therefore the overwhelming bulk of the analysis only needs to deal PTL with neither *until* nor past time. Moreover, the analysis of PTL with infinite time naturally reduces to that for PTL with just finite time. The low-level formulas also have associated practical decision procedures, including a simple symbolic one based on Binary Decision Diagrams (BDDs) [8] which we have implemented.

The basic version of PTL used here is described in detail in Sect. 3 but we will now briefly summarise some of the features in order to be able to overview some key aspects of our work. We postpone the treatment of *until* and past time in order to later handle them in a natural hierarchical manner. Both finite and infinite time are permitted, whereas most versions of PTL deal solely with the latter. One reason for including finite time is to allow us to naturally capture parts of our infinite-time analysis within PTL formulas concerning finite-time subintervals. The only two primitive temporal operators initially considered are $\bigcirc$ (strong next) and $\Diamond$ (eventually) although some others are definable in terms of them (e.g., $\Box$ (henceforth) and $\Diamond^+$ (strict eventually)).

Our analysis of PTL extensively employs intervals of time which are represented as finite and countably infinite sequences of states and described by formulas in a propositional version of Interval Temporal Logic (ITL) [29, 48–51] (see also [38]) referred to as PITL. By using a hierarchical, interval-oriented framework, the approach differs from that of Lichtenstein and Pnueli and previous ones which in general utilise sets of formulas and sequences of such sets (also referred to as *paths*). We instead relate transition configurations to semantically equivalent formulas in PITL. Time intervals facilitate an analysis which naturally relates larger intervals with smaller ones. The process of doing this can be explicitly expressed in PITL in a way not possible within previous frameworks which lack both a formalisation of intervals and logical operators concerning various kinds of sequential composition of intervals.

Let us now informally consider as an example a simplified presentation of how we later establish the existence of periodic models for certain kinds of low-level formulas involving infinite time. The analysis for temporal logic formulas involving infinite time needs to consider formulas of the form $\Box \Diamond^+ A$, where A is itself a restricted kind of temporal logic formula. Here $\Box \Diamond^+ A$ is true for an interval, that is, the interval *satisfies* $\Box \Diamond^+ A$, iff the interval has infinite length and A itself is satisfied by an infinite number of the interval's suffixes. We want to show that if $\Box \Diamond^+ A$ is satisfied by some interval, then there also exists a periodic interval which satisfies $\Box \Diamond^+ A$. We first show a sufficient condition motivated by A's restricted syntax which ensures that $\Box \Diamond^+ A$ is semantically equivalent to the PITL formula A^ω. This formula is true on an interval if the interval has infinite length and can be split into an infinite sequence of finite intervals each satisfying A. We then select one of these finite intervals and join ω copies of it together to obtain a periodic interval satisfying A^ω and hence also the original formula $\Box \Diamond^+ A$. Furthermore, after showing the existence of bounded models for A, we can then establish similar properties for A^ω and hence also $\Box \Diamond^+ A$.

We believe that our interval-based analysis complements existing approaches since it provides a notational way to articulate various issues concerning PTL model construction which are equally relevant within a more conventional analysis but are normally only considered at the metalevel. It also illustrates some general techniques for compositional specification and proof in discrete linear time which are applicable here. This all fits nicely with one of the main purposes of a logic which is to provide a notation for explicitly and formally expressing reasoning processes. In addition, a number of the temporal logic formulas encountered can even be used with little or no change as input to a implementation of a PTL decision procedure which supports both finite and infinite time. The analysis itself is performed without the need to add any fundamentally new concepts to PITL but does require a reader's willingness to acquire some familiarity with PITL and various fairly general issues concerning interval-based reasoning.

Another feature of our approach is that it readily generalises to a finite-time analysis of an important subset of PITL called *Fusion Logic* (FL), which was previously used by us in [57] to hierarchically show the completeness of an axiom system for PITL. The analysis of FL uses a reduction of FL formulas to PTL ones. The prototype implementation of our PTL decision procedure also supports FL. A brief introduction to FL is given in §13.4 since FL is a natural extension of our framework for studying PTL and furthermore demonstrates another connection between PTL and intervals. We plan in future work to give a more detailed discussion of the decision procedure for FL as well as some other issues concerning FL.

Our preliminary work in [58] contains an earlier description of this material but was limited to showing axiomatic completeness for PTL without past time. In the mean time, we have significantly extended the notation, methods and their scope of application. The structure of presentation has also been refined.

The use of intervals here seems to go well with a growing general awareness even in industry of the desirability for temporal logics which go beyond conventional point-based constructs to also handle behavioural specifications involving intervals of time. As evidence for this we mention the Property Specification Language PSL/Sugar [63]. This is a modified version of a language Sugar [3] developed at IBM/Haifa. PSL/Sugar has been ratified as IEEE standard 1850 with the purpose of precisely expressing a hardware system's design properties so that they can then be tested using simulation and model checking. It includes a temporal logic with regular expressions and other operators for sequential composition. The hardware description language SystemVerilog [66] is an extension of the established IEEE 1364-2001 Verilog language and includes temporal assertions similar to those in PSL/Sugar. SystemVerilog has itself been ratified as a standard by Accellera Organization, Inc. which also hopes to obtain ratification from the IEEE.

In addition, the IEEE Design Automation Standards Committee has recently approved a project to produce a candidate standard for Verisity Ltd.'s [68] *e* language which is intended for testing and verification[1]. A subset of *e* called *temporal e* was influenced in part by ITL [35, 47, 69]. The IEEE Standards Association has assigned the project the number 1647 [37].

Structure of Presentation

Let us now summarise the structure of the rest of this paper. Section 2 mentions some related work and gives a comparison with our approach. Section 3 presents the version of PTL we use. Section 4 summarises the propositional version of ITL which we use in the analysis. Section 5 introduces low level PTL formulas called *transition configurations* and relates them to some semantically equivalent propositional ITL formulas which simplify the subsequent analysis. Section 6 proves the existence of small models for transition configurations. Section 7 shows how to relate the satisfiability of the two main kinds of transition configurations with simple interval-oriented tests. Section 8 deals with a practical BDD-based decision procedure for transition configurations. Section 9 concerns axiomatic completeness for an important subset of PTL in which the only temporal operator is $\bigcirc$ (next). Section 10 looks at a PTL axiom system and axiomatic completeness for transition configurations. Section 11 presents formulas called *invariants* and *invariant configurations* which together serve as a bridge between the previously mentioned transition configurations and arbitrary PTL formulas. Section 12 discusses how to generalise the previous results to work with arbitrary PTL formulas. Section 13 hierarchically extends our approach to deal with both the temporal operators *until* and past time. It also briefly looks at a superset of PTL called *Fusion Logic*. Section 14 concludes with some brief discussion.

[1] Verisity has been acquired by Cadence Design Systems [11].

2 Background

Temporal logics have become a popular topic of study in theoretical computer science and are also being utilised by industry to locate faults in digital circuit designs, communication protocols and other applications. Issues such as small models, proof systems, axiomatic completeness and decision procedures for PTL (almost always limited to infinite time) have been extensively investigated by Gabbay et al. [27], Wolper [73], Kröger [40], Goldblatt [28], Lichtenstein and Pnueli [43], Lange and Stirling [42], Pucella [64] (who also considers PTL with finite time) and others. French [26] elaborates on the presentation by Gabbay et al. [27].

Vardi and Wolper [67] and Bernholtz, Vardi and Wolper [6] describe decisions procedures for some temporal logics based on a reduction to ω-automata. They do not consider axiomatic completeness. Wolper [71] presents a tutorial on such decision procedure for PTL with infinite time.

Ben-Ari et al. [4, 5], Wolper [70, 72] and Banieqbal and Barringer [2] develop closely related proofs of completeness for logics which include PTL as a subset or are branching-time versions of it. The book by Rescher and Urquhart [65] is an early source of tableau-based completeness proofs for temporal logics. The survey by Emerson [20] includes material about axiom systems for both linear and branching-time temporal logic.

Fisher [23, 24] (see also later work by Fisher, Dixon and Peim [25] and Bolotov, Fisher and Dixon [7]) presents a normal form for PTL called *Separated Normal Form* (SNF) which consists of formulas having the syntax $\Box \bigwedge_i A_i$, where each A_i can be one of the following:

$$\mathbf{start} \;\supset\; \bigvee_c l_c \qquad \bigcirc \bigwedge_a k_a \;\supset\; \bigcirc \bigvee_d l_d \qquad \bigcirc \bigwedge_b k_b \;\supset\; \Diamond\, l \ .$$

Here each particular k_a, k_b, l, l_c and l_d is a literal (i.e., a proposition variable or its negation). Some versions of SNF permit past-time constructs or have other relatively minor differences. Applications include theorem proving, executable specifications and representing ω-automata. We mention SNF here since it is a PTL normal form which somewhat resembles what we call invariants and formally introduce in Sect. 11.

3 Overview of PTL

This section summarises the basic version of PTL used here. Later on in Sect. 13 we augment PTL with the operator *until* and past time.

3.1 Syntax of PTL

We now describe the syntax of permitted PTL formulas. In what follows, p is any propositional variable and both X and Y denote PTL formulas:

$$p \qquad \mathit{true} \quad \neg X \quad X \vee Y \quad \bigcirc X \ (\text{“strong next”}) \quad \Diamond X \ (\text{“eventually”}) \ .$$

We include *true* as a primitive so as to avoid a definition of it which contains some specific variable. This is not strictly necessary. Other conventional logic operators such as *false*, $X \wedge Y$ and $X \supset Y$ (X implies Y) are defined in the usual way. Also, $\Box X$ (“henceforth”) is defined as $\neg \Diamond \neg X$.

3.2 Semantics of PTL

The version of PTL considered here uses discrete, linear time which is represented by intervals each consisting of a sequence of one or more states. More precisely, an interval σ is any finite or infinite sequence of one or more states $\sigma_0, \sigma_1, \ldots$. Each state σ_i in σ maps each propositional variable p, $q, \ldots$ to one of the boolean values *true* and *false*. The value of p in the state σ_i is denoted $\sigma_i(p)$. A finite interval σ has an *interval length* $|\sigma| \geq 0$ which equals the number of states minus 1 and is hence always greater than or equal to 0. We regard the smallest nonzero interval length 1 as a *unit* of (abstract) time. For example, an interval with 6 states has interval length 5 or equivalently 5 time units. These units do not correspond to any particular notion of physical time. The interval length of an infinite interval is taken to be ω. The term *subinterval* refers to any interval obtained from some *contiguous* subsequence of another interval's states.

$skip \wedge sfin\ \neg p$	p: t f
$\bigcirc \$(p \supset \bigcirc \neg p) \wedge \neg \$(p \wedge \bigcirc p)$	p: t t f
$\diamondsuit\!\!\!\!\!{\scriptscriptstyle m}\ p \wedge \neg \diamondsuit\!\!\!\!\!{\scriptscriptstyle m}\ \neg p \wedge \Diamond p \wedge \Diamond \neg p$	p: t t t f
$\boxed{\scriptstyle m}(p \supset \bigcirc \neg p) \wedge \neg \Box(p \supset \bigcirc \neg p)$	p: t f t f f t
$\boxed{\scriptstyle m}(p \supset \Diamond \neg p) \wedge sfin\ p$	p: t t t t f t

Figure 1: Some examples of formulas with derived PTL operators

We call a one-state interval (i.e., interval length 0) an *empty interval.* A two-state interval (i.e., interval length 1) is called a *unit interval.* Both kinds of intervals play an important role in our analysis.

The notation $\sigma \models X$ denotes that the interval σ *satisfies* the PTL formula X. We now give a definition of this using induction on X's syntax:

- Propositional variable: $\sigma \models p$ iff p is true in the initial state σ_0 (i.e., $\sigma_0(p) = true$).
- True: $\sigma \models true$ trivially holds for any σ.
- Negation: $\sigma \models \neg X$ iff $\sigma \not\models X$.
- Disjunction: $\sigma \models X \vee Y$ iff $\sigma \models X$ or $\sigma \models Y$.
- Next: $\sigma \models \bigcirc X$ iff $\sigma' \models X$,
  where σ contains at least two states and σ' denotes the suffix subinterval $\sigma_1\sigma_2\ldots$ which starts from second state σ_1 in σ.
- Eventually: $\sigma \models \Diamond X$ iff $\sigma' \models X$,
  for some suffix subinterval σ' of σ (perhaps σ itself).

Table 1 shows a variety of other useful temporal operators which are definable in PTL. It includes operators for testing whether an interval is finite or infinite and whether the interval has exactly one state or two states. Most of the operators only become relevant when finite intervals are permitted. Therefore, readers who are just familiar with conventional PTL and infinite time will have previously encountered only a few of the operators.

Note: Some readers will (quite reasonably) prefer to skim Table 1 for now and only later consult it in more detail when the various operators are actually used.

Figure 1 assists in the understanding of Table 1 by illustrating a number of the operators through sample formulas and intervals. In the figure, the logical values *true* and *false* are respectively abbreviated as "t" and "f". In what follows, we frequently use $\boxed{\scriptstyle m}$ instead of $\Box$ since we need to test pairs of adjacent states in a interval. The operator $\boxed{\scriptstyle m}$ is better suited for this since it does not "run off the end" when examining finite intervals. The fourth example in Figure 1 serves as an example of this feature. As a consequence, $\boxed{\scriptstyle m}$ is easier to work with in our interval-based analysis as is later shown in Theorem 11.

Definition 1 (Satisfiability and Validity) *For any interval σ and* PTL *formula X, if σ satisfies X (i.e., $\sigma \models X$ holds), then X is said to be* satisfiable, *denoted as $\models\!\!\!\dashv X$. A formula X satisfied by all intervals is* valid, *denoted as $\models X$.*

Standard derived PTL *operators:*		
$\Box X$	$\stackrel{\text{def}}{\equiv} \neg \Diamond \neg X$	Henceforth
$\Diamond^{+} X$	$\stackrel{\text{def}}{\equiv} \bigcirc \Diamond X$	Eventually in strict future
$\Box^{+} X$	$\stackrel{\text{def}}{\equiv} \neg \Diamond^{+} \neg X$	Henceforth in strict future (not used here)
PTL *operators primarily for finite intervals:*		
$more$	$\stackrel{\text{def}}{\equiv} \bigcirc\, true$	More than one state
$empty$	$\stackrel{\text{def}}{\equiv} \neg more$	Only one state (*empty interval*)
$\textcircled{w} X$	$\stackrel{\text{def}}{\equiv} \neg \bigcirc \neg X$	Weak next (same as $more \supset \bigcirc X$)
$skip$	$\stackrel{\text{def}}{\equiv} \bigcirc\, empty$	Exactly two states (*unit interval*)
$X?$	$\stackrel{\text{def}}{\equiv} X \wedge empty$	Empty interval with test
$\$\, X$	$\stackrel{\text{def}}{\equiv} X \wedge skip$	Unit interval with test
PTL *operators for finite and infinite intervals:*		
$finite$	$\stackrel{\text{def}}{\equiv} \Diamond\, empty$	Finite interval
inf	$\stackrel{\text{def}}{\equiv} \neg finite$	Infinite interval
$sfin\ X$	$\stackrel{\text{def}}{\equiv} \Diamond(empty \wedge X)$	Strong test of final state
$fin\ X$	$\stackrel{\text{def}}{\equiv} \Box(empty \supset X)$	Weak test of final state
$\Diamond\!\!\!\!\!{\scriptscriptstyle m}\;\, X$	$\stackrel{\text{def}}{\equiv} \Diamond(more \wedge X)$	Sometime before the very end
$\Box\!\!\!\!\!{\scriptscriptstyle m}\;\, X$	$\stackrel{\text{def}}{\equiv} \Box(more \supset X)$	Henceforth except perhaps at very end

Table 1: Some definable PTL operators

We now define an important subset of PTL involving the operator $\bigcirc$:

Definition 2 (Next Logic) *The set of* PTL *formulas in which the only primitive temporal operator is* $\bigcirc$ *is called* Next-Logic (NL). *The subset of* NL *in which no* $\bigcirc$ *is nested within another* $\bigcirc$ *is denoted as* NL^1.

For example, the NL formula $p \wedge \bigcirc q$ is in NL^1, whereas the NL formula $p \wedge \bigcirc(q \vee \bigcirc p)$ is not.

The variables T, T' and T'' denote formulas in NL^1.

Definition 3 (Tautologies) *A* tautology *is any formula which is a substitution instance of some valid nonmodal propositional formula.*

For example, the formula $\bigcirc X \vee \Diamond Y \supset \Diamond Y$ is a tautology since it is a substitution instance of the valid nonmodal formula $\models p \vee q \supset q$. It is not hard to show that all tautologies are themselves valid since intuitively a tautology is any valid formula which does not require modal reasoning to justify its truth.

Convention for variables denoting individual formulas and sets of formulas: In what follows, the variable w refers to a *state formula*, that is, a formula with no temporal operators. Furthermore, PROP denotes the set of all state formulas. For any finite set of variables V, PROP_V denotes the set of all state formulas only having variables in V. Likewise, the set PTL_V denotes the set of all formulas in PTL only containing variables in V and NL^1_V denotes the set of all formulas in NL^1 only having variables in V. For example, the formula $p \wedge \Diamond q$ is in $\mathrm{PTL}_{\{p,q\}}$ but not in $\mathrm{PTL}_{\{p\}}$.

3.3 Example of the Hierarchical Process

Our analysis of PTL reduces arbitrary PTL formulas to lower level ones with a much more restricted syntax. The next PTL formula serves as a simple example to motivate some of the notation and conventions later introduced:

$$\Box \Diamond p \wedge \Box \Diamond \neg p \ .$$

This is reducible to the formula $\Box I \wedge w$, where I and w are given below:

$$\begin{array}{ll} I\colon & (r_1 \equiv \Diamond p) \wedge (r_2 \equiv \Diamond \neg r_1) \wedge (r_3 \equiv \Diamond \neg p) \wedge (r_4 \equiv \Diamond \neg r_3) \\ w\colon & \neg r_2 \wedge \neg r_4 \ . \end{array}$$

The auxiliary variables $r_1, \ldots, r_4$ provide a natural way to eliminate the nesting of temporal operators within other temporal operators in I. We call the conjunction I an *invariant* and the conjunction $\Box I \wedge w$ an *invariant configuration*. Both are formally introduced later in Sect. 11. It can be shown that the original formula $\Box \Diamond p \wedge \Box \Diamond \neg p$ is satisfiable iff the invariant configuration $\Box I \wedge w$ is.

When analysing behaviour in finite time, we further transform the invariant configuration $\Box I \wedge w$ to another special kind of conjunction $\Box T \wedge w \wedge \mathit{finite}$, where T and w are as follows:

$$\begin{array}{ll} T\colon & (r_1 \equiv (p \vee \bigcirc r_1)) \wedge (r_2 \equiv (\neg r_1 \vee \bigcirc r_2)) \\ & \wedge (r_3 \equiv (\neg p \vee \bigcirc r_3)) \wedge (r_4 \equiv (\neg r_3 \vee \bigcirc r_4)) \\ w\colon & \neg r_2 \wedge \neg r_4 \ . \end{array}$$

Here I's first conjunct $r_1 \equiv \Diamond p$ is replaced in T by the $\Diamond$-free formula $r_1 \equiv (p \vee \bigcirc r_1)$. The remaining conjuncts in T similarly avoid having any $\Diamond$ constructs. We call T a *transition formula* and $\Box T \wedge w \wedge \mathit{finite}$ a *transition configuration* (formally defined in Section 5). The formula T is in fact a formula in the important subset of PTL called NL^1 (previous formally defined in Definition 2) in which the only temporal constructs are $\bigcirc$ operators not nested within other $\bigcirc$ operators. In addition, in finite-time intervals the PTL formulas $\Box I$ and $\Box T$ are semantically equivalent. Moreover, it can be shown that the original formula $\Box \Diamond p \wedge \Box \Diamond \neg p$ is satisfiable in finite time iff the transition configuration $\Box T \wedge w \wedge \mathit{finite}$ is satisfiable. As is later shown in Sect. 5, NL^1 formulas such as T play a fundamental role in our analysis of transition configurations.

3.4 Notation for Accessing Parts of Conjunctions

From the examples just given it can be seen that we often manipulate formulas which are conjunctions. The next three definitions provide some helpful notation for denoting the number of conjuncts of such a formula and for accessing one or more of them.

Definition 4 (Size of a Conjunction) *For any conjunction C of zero or more conjuncts, let the notation $|C|$ denote the number of C's conjuncts.*

Definition 5 (Indexing of a Conjunction's Conjuncts) *For each $k : 1 \leq k \leq |C|$, we let $C[k]$ denote the k-th conjunct.*

Observe that if a conjunction C has length $|C| = 0$, there are no conjuncts to be indexed.

Definition 6 (Parts of a Conjunction) *Suppose C is a conjunction and k and l are natural numbers such that $1 \leq k \leq |C|$ and $0 \leq l \leq |C|$. The notation $C[k : l]$ denotes the conjunction of consecutive conjuncts in C starting with $C[k]$ and finishing with $C[l]$, inclusive, i.e., $C[k] \wedge \cdots \wedge C[l]$ (which contains $l - k + 1$ conjuncts).*

Note that for any conjunction C, the formula $C[1 : 0]$ denotes *true* and $C[1 : |C|]$ is identical to C. Also, for any $k : 1 \leq k \leq |C|$, both $C[k]$ and $C[k : k]$ refer to the same conjunct.

4 Propositional Interval Temporal Logic

We now describe the version of quantifier-free propositional ITL (PITL) used here for systematically analysing transition configurations. More on ITL can be found in [29, 48–52, 55–57] (see also [38]). The same discrete-time intervals are used as in PTL. In addition, all PTL constructs are permitted as well as two other ones. Hence, any PTL formula is also a PITL formula.

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

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

The semantics of the other constructs in PITL is as in PTL and is therefore omitted here.

Before defining the semantics of chop and chop-star, we introduce some notation for describing subintervals of an interval σ. For natural numbers i, j with $i \leq j \leq |\sigma|$, let $\sigma_{i:j}$ denotes the subinterval with starting state σ_i and final state σ_j and having interval length $j - i$ (i.e., $j - i + 1$ states). Furthermore, if σ is an infinite interval, let $\sigma_{i:\omega}$ denote the (infinite) suffix subinterval starting with state σ_i.

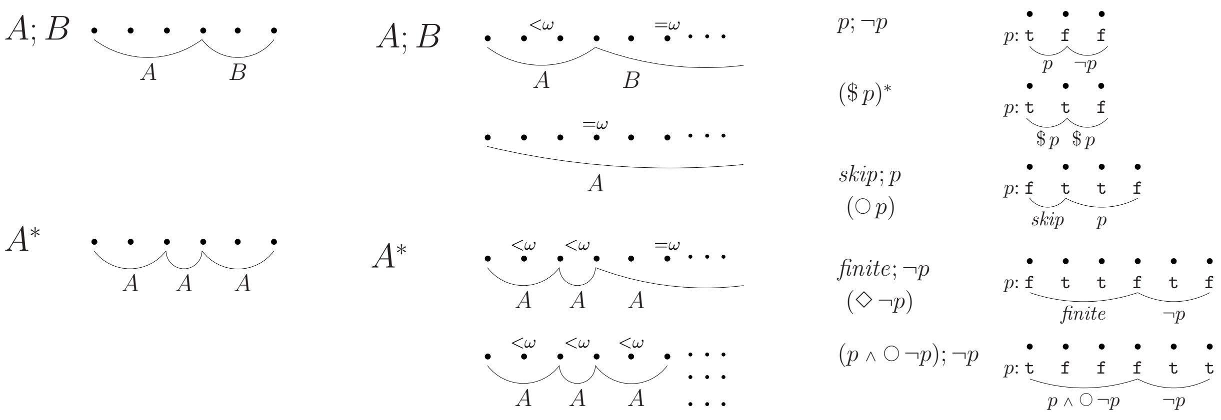
The formula $A; B$ is true on σ (i.e., $\sigma \models A; B$) iff one of the following holds:

- For some natural number $i : 0 \leq i \leq |\sigma|$, the interval σ can be divided into two subintervals $\sigma_{0:i}$ and $\sigma_{i:|\sigma|}$ sharing the state σ_i such that both $\sigma_{0:i} \models A$ and $\sigma_{i:|\sigma|} \models B$ hold.
- The interval σ itself has infinite length and $\sigma \models A$ holds.

The formula A^* is true on σ (i.e., $\sigma \models A^*$) iff one of the following holds:

- The interval σ has finite length and there exists some natural number $n \geq 0$ and finite sequence of natural numbers $l_0 \leq l_1 \leq \cdots \leq l_n$ where $l_0 = 0$ and $l_n = |\sigma|$, such that for each $i : 0 \leq i < n$, $\sigma_{l_i:l_{i+1}} \models A$ holds.

  The behaviour of chop-star on empty intervals is a frequent source of confusion and it is therefore important to note that any formula A^* (including *false**) is true on a one-state interval. This is because in the semantics of chop-star for a one-state interval we can always set $n = 0$ and therefore ignore the values of variables in the interval σ.
- The interval σ has infinite length and there exists some $n \geq 0$ and finite sequence of natural numbers $l_0 \leq l_1 \leq \cdots \leq l_n$ where $l_0 = 0$, such that for each $i : 0 \leq i < n$, $\sigma_{l_i:l_{i+1}} \models A$ holds and also $\sigma_{l_n:\omega} \models A$ holds.



(a) Informal semantics for finite time

(b) Informal semantics for infinite time

(c) Some finite-time examples

Figure 2: Informal PITL semantics and examples

$\Diamond_i\, A$	$\stackrel{\text{def}}{\equiv}$	$A; \mathit{true}$	A is true in some initial subinterval
$\boxed{\text{i}}\, A$	$\stackrel{\text{def}}{\equiv}$	$\neg \Diamond_i \neg A$	A is true in all initial subintervals
$\Diamond_a\, A$	$\stackrel{\text{def}}{\equiv}$	$\mathit{finite}; A; \mathit{true}$	A is true in some subinterval
$\boxed{\text{a}}\, A$	$\stackrel{\text{def}}{\equiv}$	$\neg \Diamond_a \neg A$	A is true in all subintervals

Table 2: Some useful derived PITL operators

- The interval σ has infinite length and there exists some countably infinite strictly ascending sequence of natural numbers $l_0 < l_1 < \cdots$ where $l_0 = 0$, such that for each $i : i \geq 0$, $\sigma_{l_i:l_{i+1}} \models A$ holds.

Figure 2 pictorially illustrates the semantics of *chop* and *chop-star* in both finite and infinite time and also shows some simple PITL formulas together with intervals which satisfy them. For some sample formulas we include in parentheses versions using conventional PTL logic operators which were previously introduced in Sect. 3.

We make use of the following definitions of two straightforward forms of iteration expressible with chop and chop-star:

$$A^{+} \stackrel{\text{def}}{\equiv} A; A^{*} \qquad A^{\omega} \stackrel{\text{def}}{\equiv} (A \wedge \mathit{finite})^{*} \wedge \mathit{inf} \quad .$$

In addition, for any $n \geq 0$, we define A^n to be the formula *empty* if $n = 0$ and otherwise to be $A; A^{n-1}$. The constructs $A^{\leq n}$ and $A^{<n}$ are defined to be the disjunctions $\bigvee_{k \leq n} A^k$ and $\bigvee_{k<n} A^k$, respectively.

Other derived operators are also possible. Table 2 shows some especially useful ones.

The notions of satisfiability and validity already introduced in Definition 1 for PTL naturally generalise to PITL.

Let PITL_V be the set of all PITL formulas only having variables in V.

The next definition introduces a special kind of state formula which is indispensable for interval-based reasoning. It plays the role that sets of formulas typically do in other analyses of PTL.

Definition 7 (Atoms and V-Atoms) *An* atom *is any finite conjunction in which each conjunct is some propositional variable or its negation and no two conjuncts share the same variable. The set of all atoms is denoted Atoms. The Greek letters α, β and γ denote individual atoms. For any finite set of propositional* *variables V, let $Atoms_V$ be some set of $2^{|V|}$ logically distinct atoms containing exactly the variables in V. We refer to such atoms as V-*atoms*.*

For example, we can let $Atoms_{\{p,q\}}$ be the set of the four logically distinct atoms shown below:

$$p \wedge q \qquad p \wedge \neg q \qquad \neg p \wedge q \qquad \neg p \wedge \neg q \ .$$

One simple convention is to assume that the propositional variables in an atom occur from left to right in lexical order. For any finite set of variables V, this immediately leads to a suitable set of $2^{|V|}$ different V-atoms.

5 Transition Configurations

Starting with a finite set of variables V, an NL^1_V formula T and a state formula *init* in PROP_V, we consider small models, a decision procedure and axiomatic completeness for certain low-level formulas referred to here as *transition configurations*. These formulas play a central role in our approach. The analysis of arbitrary PTL formulas can be ultimately reduced to that of transition configurations.

Before actually formally defining transition configurations, we need to introduce the concept of a *conditional liveness formula* which is a specific kind of conjunction necessary for reasoning about liveness properties involving infinite time. The definition therefore makes use of some general notation already introduced in Definitions 4–6 for manipulating conjunctions.

Definition 8 (Conditional Liveness Formulas) *A* conditional liveness formula *L is a conjunction of $|L|$ implications $L[1] \wedge \cdots \wedge L[|L|]$. Each implication has the form $w \supset \Diamond\!\!\!\!\cdot\;\, w'$, where w and w' are two state formulas. For convenience, we let $\eta_{L[k]}$ denote the left operand of the k-th implication in L. Similarly, $\theta_{L[k]}$ denotes the operand of the $\Diamond\!\!\!\!\cdot\;\,$ formula in the k-th implication $L[k]$'s right side. Therefore, for each $k : 1 \leq k \leq |L|$, the implications $L[k]$ and $\eta_{L[k]} \supset \Diamond\!\!\!\!\cdot\;\, \theta_{L[k]}$ denote the same formula.*

For any V-atom α and any $k : 1 \leq k \leq |L|$, if the formula $\alpha \wedge \eta_{L[k]}$ is satisfiable, we say that α enables *L's k-th implication $L[k]$.*

Here is a sample conditional liveness formula:

$$((p \vee \neg q) \supset \Diamond\!\!\!\!\cdot\;\, \neg p) \quad \wedge \quad (q \supset \Diamond\!\!\!\!\cdot\;\,(p \equiv \neg q)) \quad \wedge \quad (true \supset \Diamond\!\!\!\!\cdot\;\,(p \supset q)) \ . \tag{1}$$

Note that $\Diamond\!\!\!\!\cdot\;\,$ behaves the same as $\Diamond$ on infinite intervals. However, in finite intervals $\Diamond\!\!\!\!\cdot\;\,$, like its dual $\boxdot$, ignores the final state. In principle, either $\Diamond\!\!\!\!\cdot\;\,$ or $\Diamond$ can be used in conditional liveness formulas and the choice between them appears to be largely a matter of taste. Nevertheless, we choose to use $\Diamond\!\!\!\!\cdot\;\,$ in part because it facilitates an interesting generalisation of both conditional liveness formulas and another kind of formula called an *invariant* which is introduced later in Sect. 11. This generalisation will be mentioned in §13.3. In addition, the application of $\Diamond\!\!\!\!\cdot\;\,$ naturally complements our extensive use of its dual $\boxdot$.

Here is the definition of transition configurations:

Definition 9 (Transition Configurations) *A* transition configuration *is a formula of the form $\Box T \ \wedge \ X$, where the formula T is in NL^1_V, and the PTL_V formula X has one of the four forms shown below:*

Type of transition configuration	*Syntax of X*
Finite-time	$init \wedge finite$
Infinite-time	$init \wedge \Box \Diamond^+ L$
Final	$w \wedge empty$
Periodic	$\alpha \wedge L \wedge \Box \Diamond^+(\alpha \wedge L)$

Here init is a state formula in PROP_V which corresponds to some initial *condition, w is some state formula in PROP_V, L is a conditional liveness formula in PTL_V and α is a V-atom. If init is the formula true, it can be omitted. The same applies with w.*

For example, the conjunction $\Box(more \supset (p \equiv \bigcirc p)) \ \wedge \ p \ \wedge \ finite$ is a finite-time transition configuration which is true exactly for finite intervals in which p is always true.

Type of transition configuration	PITL_V formula	Where proved
Finite-time	$((\$T)^* \wedge \mathit{init} \wedge \mathit{finite}); (T \wedge \mathit{empty})$	Theorem 17
Infinite-time	$((\$T)^* \wedge \mathit{init} \wedge \mathit{finite});$ $\big((\$T)^* \wedge L \wedge (\vec{V} \leftarrow \vec{V})\big)^\omega$	Theorem 26
Final	$T \wedge w \wedge \mathit{empty}$	straightforward
Periodic	$((\$T)^* \wedge \alpha \wedge L)^\omega$	Theorem 24

Table 3: Reduction of transition configurations to PITL_V formulas

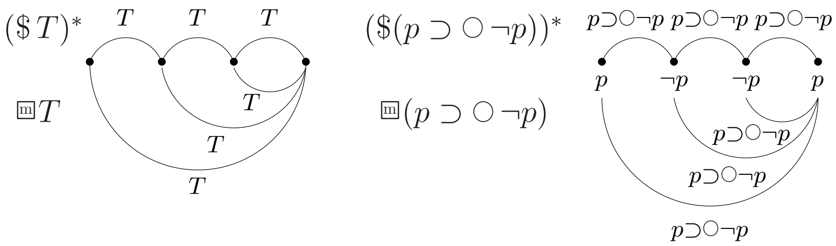


Figure 3: Illustration of equivalence of $(\$T)^*$ and $\boxdot T$

Note: In the course of analysing transition configurations, we will assume that V, T, *init* and L are fixed.

We will show that finite-time and infinite-time transition configurations are equivalent to certain PITL_V formulas for which we can more readily establish such things as the existence of periodic models, small models, a decision procedure and axiomatic completeness. Table 3 shows the corresponding PITL_V formula for each kind of transition configuration and where the equivalence of the two is proved. Here $\vec{V} \leftarrow \vec{V}$ denotes that the initial value of each variable occurring in the set of variables V equals its final value. It can be expressed as the PTL_V formula $\mathit{finite} \supset \bigwedge_{v \in V}(v \equiv \mathit{fin}\ v)$ and is semantically equivalent to the disjunction $\bigvee_{\alpha \in \mathit{Atoms}_V}(\alpha \wedge \mathit{fin}\ \alpha)$.

Theorem 36 will furthermore establish that the infinite-time transition configuration is satisfiable iff the next PTL formula is satisfiable in finite time:

$$\boxdot T \wedge \mathit{init} \wedge \Diamond(L \wedge \mathit{finite} \wedge \mathit{more} \wedge (\vec{V} \leftarrow \vec{V})) \ .$$

In order to perform interval-based analysis on transition configurations, we need to relate $\Box T$ to the PITL formula $(\$T)^*$. Now the PTL formula $\boxdot T$, which is very similar to $\Box T$, was previously defined in Table 1 to be true on an interval iff T is true in all of the interval's nonempty suffix subintervals. It turns out that due to T being in NL^1, the formula $(\$T)^*$ is semantically equivalent to $\boxdot T$. Intuitively, this is because an NL^1 formula cannot probe past the second state of an interval. The next lemma formalises this:

Lemma 10 *Let σ and σ' be two nonempty intervals which share the same first two states (i.e., $\sigma_0 = \sigma'_0$ and $\sigma_1 = \sigma'_1$). Then, for any formula T in NL^1, σ satisfies T iff σ' satisfies T.*

Proof Induction on T's syntax ensures that it cannot distinguish between σ and σ'. $\Box$

Consequently, if two nonempty intervals share the same first two states, then the truth value of T for both intervals is identical. Figure 3 illustrates this with two instances of an interval containing 4 states. The second version uses the concrete NL^1 formula $p \supset \bigcirc\neg p$ and shows specific values for the proposition variable p. Both $(\$T)^*$ and $\boxdot T$ test each pair of adjacent states. The equivalence consequently permits us to express $(\$T)^*$ in PTL by means of $\boxdot T$. In addition, it is often useful to express $\boxdot T$ as $(\$T)^*$ because the later turns out to be much more suitable for interval-based reasoning involving sequential composition and decomposition.

We now formally establish the semantic equivalence of the formulas $(\$T)^*$ and $\boxdot T$:

Theorem 11 *The* PITL *formula* $(\$ T)^*$ *and the* PTL *formula* $\boxed{\scriptstyle m}\, T$ *are semantically equivalent and hence the equivalence* $(\$ T)^* \equiv \boxed{\scriptstyle m}\, T$ *is valid.*

Proof Given an interval σ, we can put each two-state (unit) subinterval in one-to-one correspondence with the suffix (nonempty) subinterval which shares the first two states. Now σ satisfies $(\$ T)^*$ iff T is true on all of σ's unit subintervals. Similarly, σ satisfies $\boxed{\scriptstyle m}\, T$ iff T is true on all of σ's nonempty suffix subintervals. By the previous Lemma 10 a given unit subinterval satisfies T iff the matching suffix (nonempty) subinterval satisfies T. Consequently, the overall interval satisfies $(\$ T)^*$ iff it satisfies $\boxed{\scriptstyle m}\, T$. $\Box$

It is not hard to check that on a one-state (empty) interval, $\boxed{\scriptstyle m}\, T$ is trivially true. On a two-state (unit) interval, it is semantically equivalent to the formula T itself.

Also note that the PTL formula $\Box T$ is semantically equivalent to the PTL formula $\boxed{\scriptstyle m}\, T \wedge \mathit{fin}\ T$. This fact and Theorem 11 together establish that $\Box T$ is also semantically equivalent to the PITL formula $(\$ T)^* \wedge \mathit{fin}\ T$. Therefore, the $\Box T$ formula in transition configurations can be readily re-expressed in PITL as the conjunction $(\$ T)^* \wedge \mathit{fin}\ T$. This will assist our interval-based analysis of transition configurations.

Remark 12 *We have discussed the important semantic equivalence of the formulas* $(\$ T)^*$ *and* $\boxed{\scriptstyle m}\, T$ *with quite a few people who themselves have a considerable amount of experience with both* PTL *and* PITL. *Originally we thought that this amounted to a straightforward application of temporal logic. However, to our surprise, these people found the equivalence and its applications to be nontrivial and interesting. For this reason, we have designated the statement of the equivalence of* $(\$ T)^*$ *and* $\boxed{\scriptstyle m}\, T$ *to be a theorem (i.e., the previous Theorem 11), rather than merely a lemma.*

Here is a corollary of Theorem 11 for infinite time:

Corollary 13 *The two formulas* $\Box T$ *and* $(\$ T)^*$ *are semantically equivalent on infinite intervals and hence the implication* $\mathit{inf} \supset \Box T \equiv (\$ T)^*$ *is valid.*

Proof This readily follows from Theorem 11 and the semantic equivalence of $\boxed{\scriptstyle m}\, T$ and $\Box T$ on infinite intervals. $\Box$

The next two Lemmas 14 and 15 subsequently provide a basis for relating finite-time transition configurations to final ones and also for relating infinite-time transition configurations to periodic ones.

Lemma 14 *For any* PITL *formula* A*, the next equivalence is valid:*

$$\models \quad \Box T \wedge \Diamond A \;\equiv\; (\$ T)^* \wedge \Diamond(\Box T \wedge A) \ .$$

Proof We first establish the validity of the PTL formula $\Box p \;\equiv\; \boxed{\scriptstyle m}\, p \wedge \Diamond \Box p$ which itself leads to the validity of the formula $\Box p \wedge \Diamond q \;\equiv\; \boxed{\scriptstyle m}\, p \wedge \Diamond(\Box p \wedge q)$. We then substitute T into p and A into q. Finally, Theorem 11 permits us to replace $\boxed{\scriptstyle m}\, T$ by $(\$ T)^*$. $\Box$

Lemma 15 *For any state formula* w *and* PITL *formula* A*, the next equivalence is valid:*

$$\Box T \wedge w \wedge \Diamond A \quad \equiv \quad ((\$ T)^* \wedge w \wedge \mathit{finite}); (\Box T \wedge A) \ . \tag{2}$$

Proof Lemma 14 ensures that $\Box T \wedge \Diamond A$ is semantically equivalent to the conjunction $(\$ T)^* \wedge \Diamond(\Box T \wedge A)$. This is itself semantically equivalent to the next PITL formula:

$$((\$ T)^* \wedge \mathit{finite}); ((\$ T)^* \wedge \Box T \wedge A) \ .$$

Now $\Box T$ trivially implies $\boxed{\scriptstyle m}\, T$ which by Theorem 11 is semantically equivalent to $(\$ T)^*$. This consequently permits us to simplify the subformula $(\$ T)^* \wedge \Box T$ into $\Box T$ to obtain the next valid equivalence:

$$\models \quad \Box T \wedge \Diamond A \quad \equiv \quad ((\$ T)^* \wedge \mathit{finite}); (\Box T \wedge A) \ .$$

Simple temporal reasoning permits us to suitably add the state formula w to each side to obtain the validity of the formula (2). $\Box$

5.1 Analysis of Finite-Time Behaviour

The following Lemma 16 and Theorem 17 concern reducing a finite-time transition configuration to the associated semantically equivalent PITL formula in Table 3 which is easier to later analyse:

Lemma 16 *The following equivalence is valid for finite-time transition configurations and relates them to final configurations:*

$$\models \quad \Box T \wedge \mathit{init} \wedge \mathit{finite} \;\equiv\; ((\$T)^* \wedge \mathit{init} \wedge \mathit{finite}); (\Box T \wedge \mathit{empty}) \ . \tag{3}$$

Proof The formula *finite* is defined to be $\Diamond\, \mathit{empty}$. Therefore Lemma 15 ensures the validity of the equivalence (3). □

Theorem 17 builds on Lemma 16 by reducing a finite-time transition configuration to a chop formula in PITL which is even easier to analysis because its righthand operand is in NL^1:

Theorem 17 *The following equivalence is valid for finite-time transition configurations:*

$$\models \quad \Box T \wedge \mathit{init} \wedge \mathit{finite} \;\equiv\; ((\$T)^* \wedge \mathit{init} \wedge \mathit{finite}); (T \wedge \mathit{empty}) \ .$$

Proof This readily follows from Lemma 16 and the fact that in an empty interval, the formulas $\Box T$ and T are equivalent. □

Note that the PITL formula $((\$T)^* \wedge \mathit{init} \wedge \mathit{finite}); (T \wedge \mathit{empty})$ can also be expressed as the semantically equivalent PITL formulas $\mathit{init}?; ((\$T)^* \wedge \mathit{finite}); T?$ and $\mathit{init} \wedge (\$T)^* \wedge \mathit{sfin}\ T$. Each form has its benefits. We prefer $T \wedge \mathit{empty}$ over the equivalent $T?$ since some readers might get confused upon seeing the operator ? with an operand which is a temporal formula even though this is permitted in PITL.

5.2 Analysis of Infinite-Time Behaviour

We now turn to analysing infinite-time transition configurations. The first step involves relating them to periodic transition configurations. The next Lemma 18 does this:

Lemma 18 *The following equivalence is valid for infinite-time transition configurations:*

$$\begin{aligned} &\Box T \wedge \mathit{init} \wedge \Box \Diamond^+ L \\ &\equiv\ ((\$T)^* \wedge \mathit{init} \wedge \mathit{finite}); \bigvee_{\alpha \in \mathit{Atoms}_V} (\Box T \wedge \alpha \wedge L \wedge \Box \Diamond^+ (\alpha \wedge L)) \ . \end{aligned} \tag{4}$$

Proof Observe that in an infinite interval if L is always eventually true then for at least one of the finite number of V-atoms, the conjunction $\alpha \wedge L$ is also always eventually true. Therefore simple temporal reasoning yields that $\Box \Diamond^+ L$ is semantically equivalent to the disjunction $\bigvee_{\alpha \in \mathit{Atoms}_V} \Box \Diamond^+ (\alpha \wedge L)$. The subformula $\Box \Diamond^+ (\alpha \wedge L)$ can be re-expressed as $\Diamond(\alpha \wedge L \wedge \Box \Diamond^+ (\alpha \wedge L))$ so the next equivalence concerning $\Box \Diamond^+ L$ is valid:

$$\models \quad \Box \Diamond^+ L \;\equiv\; \Diamond \textstyle\bigvee_{\alpha \in \mathit{Atoms}_V} (\alpha \wedge L \wedge \Box \Diamond^+ (\alpha \wedge L)) \ . \tag{5}$$

We then use Lemma 15 to establish the equivalence below for some arbitrary V-atom α:

$$\begin{aligned} \models \quad &\Box T \wedge \mathit{init} \wedge \Diamond(\Box T \wedge \alpha \wedge L \wedge \Box \Diamond^+ (\alpha \wedge L)) \\ &\equiv\ ((\$T)^* \wedge \mathit{init} \wedge \mathit{finite}); (\Box T \wedge \alpha \wedge L \wedge \Box \Diamond^+ (\alpha \wedge L)) \ . \end{aligned}$$

Some simple temporal reasoning involving chop and $\bigvee$ yields the next valid equivalence:

$$\begin{aligned} \models\ &\Box T \wedge \mathit{init} \wedge \Diamond \textstyle\bigvee_{\alpha \in \mathit{Atoms}_V} (\Box T \wedge \alpha \wedge L \wedge \Box \Diamond^+ (\alpha \wedge L)) \\ &\equiv\ ((\$T)^* \wedge \mathit{init} \wedge \mathit{finite}); \textstyle\bigvee_{\alpha \in \mathit{Atoms}_V} (\Box T \wedge \alpha \wedge L \wedge \Box \Diamond^+ (\alpha \wedge L)). \end{aligned} \tag{6}$$

The combination of this and the previously mentioned semantic equivalence (5) establishes the validity of the equivalence (4). □

5.2.1 Reduction using Chop-Omega Operator

Much of the remainder of the analysis consists of showing how to further reduce a periodic transition configuration $\Box T \wedge \alpha \wedge L \wedge \Box \Diamond^{+}(\alpha \wedge L)$ to the semantically equivalent PITL formula $((\$\,T)^{*} \wedge \alpha \wedge L)^{\omega}$. A general class of formulas which includes $\alpha \wedge L$ will now be described. For any PITL formula A in this class, the two formulas $A \wedge \Box \Diamond^{+} A$ and A^{ω} will be shown to be semantically equivalent in Theorem 23. We first need to introduce a derived PITL operator which turns out to be useful for analysing periodic behaviour in infinite intervals.

Definition 19 (The Operator $\Diamond_{\!f}$) *For any* PITL *formula A, let the* PITL *formula $\Diamond_{\!f} A$ is defined to be $(A \wedge \mathit{finite}); \mathit{true}$. Therefore, $\Diamond_{\!f} A$ true on an interval iff A is true on some finite subinterval starting at the beginning of the overall interval.*

Note that $\Diamond_{\!f} A$ can also be expressed with the derived operator $\Diamond_{\!i}$ (itself previously defined in Table 2) as $\Diamond_{\!i}(A \wedge \mathit{finite})$.

It is worthwhile to define a notion of fixpoints of the operator $\Diamond_{\!f}$:

Definition 20 (Fixpoints of the Operator $\Diamond_{\!f}$) *A* PITL *formula A is a fixpoint of $\Diamond_{\!f}$ iff the equivalence $A \equiv \Diamond_{\!f} A$ is valid.*

Fixpoints of $\Diamond_{\!f}$ are easier to move out of subintervals than are arbitrary formulas. Incidentally, for any PITL formula A, the formula $\Diamond_{\!f} A$ is a trivial fixpoint of $\Diamond_{\!f}$ since $\Diamond_{\!f} A$ and $\Diamond_{\!f} \Diamond_{\!f} A$ are semantically equivalent. We will shortly show that all conditional liveness formulas are $\Diamond_{\!f}$-fixpoints and later use this in the analysis of infinite intervals.

We extensively investigate fixpoints of various temporal operators and their application to compositional reasoning in [52–55].

The next lemma characterises a broad syntactic class of formulas which are $\Diamond_{\!f}$-fixpoints and is easy to check:

Lemma 21 *Every state formula is a $\Diamond_{\!f}$-fixpoint. Furthermore, if the* PITL *formulas A and B are $\Diamond_{\!f}$-fixpoints, then so are the* PITL *formulas $A \wedge B$, $A \vee B$, $\bigcirc A$ and $\Diamond A$.*

Lemma 22 *Every conditional liveness formula is a $\Diamond_{\!f}$-fixpoint.*

Proof A conditional liveness formula is a conjunction of implications each which has the form $w \supset \Diamond_{\!m} w'$ for some state formulas w and w'. If we replace $\supset$ and $\Diamond_{\!m}$ by their definitions, then the implication reduces to the formula $\neg w \vee \Diamond((\bigcirc\, \mathit{true}) \wedge w')$. Lemma 21 then ensures that this is a $\Diamond_{\!f}$-fixpoint. Consequently, the original implication $w \supset \Diamond_{\!m} w'$ is one as well. Therefore by Lemma 21, the conjunction of such implications which constitutes a conditional liveness formula is also a $\Diamond_{\!f}$-fixpoint. $\Box$

Observe that by Lemmas 21 and 22, the formula $\alpha \wedge L$ is itself a $\Diamond_{\!f}$-fixpoint because both α and L are $\Diamond_{\!f}$-fixpoints.

Now the formula $\alpha \wedge L \wedge \Box \Diamond^{+}(\alpha \wedge L)$ is itself an instance of the PITL formula $A \wedge \Box \Diamond^{+} A$. We now proof in Theorem 23 that if A is a $\Diamond_{\!f}$-fixpoint, then the formula $A \wedge \Box \Diamond^{+} A$ can be re-expressed as the semantically equivalent PITL formula A^{ω}. This will let us re-express $\alpha \wedge L \wedge \Box \Diamond^{+}(\alpha \wedge L)$ as the semantically equivalent PITL formula $(\alpha \wedge L)^{\omega}$. The establishment of this equivalence is a key step in the reduction of reasoning about infinite time behaviour to finite time behaviour and consequently proving the existence of periodic models for satisfiable periodic transition configurations.

Theorem 23 *For any* PITL *formula A which is a $\Diamond_{\!f}$-fixpoint, the next equivalence is valid:*

$$\models \quad A \wedge \Box \Diamond^{+} A \quad \equiv \quad A^{\omega} \ . \tag{7}$$

Proof *Left side implies right side:* Suppose that an interval σ satisfies $A \wedge \Box \Diamond^{+} A$. Now this conjunction is semantically equivalent to the formula $\Diamond_{\!f} A \wedge \Box \Diamond^{+} \Diamond_{\!f} A$ because A is a $\Diamond_{\!f}$-fixpoint. Therefore σ also satisfies the formula $\Diamond_{\!f} A \wedge \Box \Diamond^{+} \Diamond_{\!f} A$. Furthermore, σ is clearly an infinite interval due to the conjunct containing $\Box \Diamond^{+}$. Therefore, σ has an infinite number of finite subintervals which all satisfy A including some starting with σ's first state. An infinite sequence of nonoverlapping finite-length subintervals all satisfying A can then be selected with the first one commencing at σ's first state. Consequently, σ satisfies the PITL formula $((A \wedge \mathit{finite}); \mathit{true})^\omega$ which is the same as $(\diamondsuit A)^\omega$. This and the assumption that A is a $\diamondsuit$-fixpoint yield that σ satisfies A^ω.

Right side implies left side: Suppose that an interval σ satisfies A^ω. Therefore σ is an infinite interval and has an infinite number of finite subintervals all satisfying A, including one starting with σ's initial state. From this we can readily obtain the valid PITL implication shown below:

$$\models \quad A^\omega \quad \supset \quad (A \wedge \mathit{finite}); \mathit{true} \;\;\wedge\;\; \Box \Diamond^+((A \wedge \mathit{finite}); \mathit{true}) \;\; .$$

This can be re-expressed using $\diamondsuit$ as follows:

$$\models \quad A^\omega \quad \supset \quad \diamondsuit A \;\wedge\; \Box \Diamond^+ \diamondsuit A \;\; . \qquad \Box$$

The assumption that A is a $\diamondsuit$-fixpoint then yields the desired validity of the semantically equivalent implication $A^\omega \;\supset\; A \wedge \Box \Diamond^+ A$.

The next Theorem 24 relates any periodic transition configuration with its associated PITL formula shown in Table 3:

Theorem 24 *The next equivalence concerning a periodic transition configuration is valid:*

$$\models \quad \Box T \wedge \alpha \wedge L \wedge \Box \Diamond^+(\alpha \wedge L) \;\;\equiv\;\; ((\$\,T)^* \wedge \alpha \wedge L)^\omega \;\; . \tag{8}$$

Proof Lemmas 21 and 22 ensure that the formula $\alpha \wedge L$ is itself a $\diamondsuit$-fixpoint because both α and L are $\diamondsuit$-fixpoints. Therefore Theorem 23 yields the validity of the equivalence $\alpha \wedge L \wedge \Box \Diamond^+(\alpha \wedge L) \equiv (\alpha \wedge L)^\omega$. Now we conjoin $\Box T$ to each side of the equivalence. We then use the fact that $\Box T$ and $(\$\,T)^*$ are semantically equivalent in infinite time (Corollary 13) so the equivalence below is valid:

$$\models \quad \Box T \wedge \alpha \wedge L \wedge \Box \Diamond^+(\alpha \wedge L) \;\;\equiv\;\; (\$\,T)^* \wedge (\alpha \wedge L)^\omega \;\; .$$

Now $(\$\,T)^* \wedge (\alpha \wedge L)^\omega$ is an instance of the PITL formula $(\$\,B)^* \wedge C^\omega$ which itself is semantically equivalent to $((\$\,B)^* \wedge C)^\omega$. The intuition here is that both of them use $\$\,B$ to test exactly all the two-state subintervals of the overall interval. Finally, we use this to re-express $(\$\,T)^* \wedge (\alpha \wedge L)^\omega$ as $((\$\,T)^* \wedge \alpha \wedge L)^\omega$, thereby obtaining the validity of formula (8). $\Box$

The following Lemma 25 concerning a disjunction of periodic transition configurations is needed to justify our reduction of the satisfiability of a infinite-time transition configuration to the associated PITL_V formula shown in Table 3:

Lemma 25 *The next equivalence is valid:*

$$\models \quad \bigvee_{\alpha \in Atoms_V} (\Box T \;\wedge\; \alpha \;\wedge\; L \;\wedge\; \Box \Diamond^+(\alpha \;\wedge\; L)) \quad\equiv\quad \big((\$\,T)^* \;\wedge\; L \;\wedge\; (\vec{V} \;\leftarrow\; \vec{V})\big)^\omega \;\; . \tag{9}$$

Proof Theorem 24 ensures that the equivalence given below is valid:

$$\models \quad \Box T \wedge \alpha \wedge L \wedge \Box \Diamond^+(\alpha \wedge L) \quad\equiv\quad ((\$\,T)^* \wedge \alpha \wedge L)^\omega \;\; .$$

Simple temporal reasoning establishes that the equivalence's righthand operand $((\$\,T)^* \wedge \alpha \wedge L)^\omega$ can then be re-expressed as the formula $\alpha \wedge ((\$\,T)^* \wedge L \wedge (\vec{V} \leftarrow \vec{V}))^\omega$. Some further simple reasoning about the operator $\bigvee$ yields the validity of the equivalence (9). $\Box$

The equivalence of an infinite-time transition configuration with the associated PITL_V formula shown in Table 3 is now established:

Theorem 26 *The following equivalence is valid for infinite-time transition configurations:*

$$\models \quad \Box T \;\wedge\; \mathit{init} \;\wedge\; \Box \Diamond^+ L \quad\equiv\quad ((\$\,T)^* \;\wedge\; \mathit{init} \;\wedge\; \mathit{finite}); \big((\$\,T)^* \;\wedge\; L \;\wedge\; (\vec{V} \;\leftarrow\; \vec{V})\big)^\omega \;\; .$$

Proof This readily follows from Lemma 18 which relates infinite-time transition configurations to periodic transition configurations and Lemma 25 which re-expresses the disjunction of several periodic transition configurations using chop-omega. $\Box$

5.2.2 Fusion and Canonical Intervals

Let us consider some general concepts and techniques concerning PITL and its notion of intervals. They will be extensively used later on.

Definition 27 (Fusion) *Let σ and σ' be two intervals. The definition of the* fusion *of them, denoted $\sigma \circ \sigma'$, has two cases, depending on whether σ has finite length or not:*

- *If σ has finite length, we require that last state of σ equals the first state of σ'. The fusion of the σ with σ' is then the interval obtained by appending the two intervals together so as to include only one copy of the shared state.*
- *Otherwise, the fusion is σ itself, no matter what σ' is.*

For example, suppose s_1, s_2 and s_3 are states. If σ is the interval s_1s_2 and σ' is the interval s_2s_3, then their fusion $\sigma \circ \sigma'$ equals the three-state interval $s_1s_2s_3$, rather than the four-state interval $s_1s_2s_2s_3$ which concatenation yields. Note that when σ has finite length and σ and σ' do not share the relevant state, then their fusion is undefined. If both σ and σ' are finite and compatible, then the interval $\sigma \circ \sigma'$ contains the total sum of states in σ and σ' minus one. Hence the interval length of $\sigma \circ \sigma'$ equals the sum of the interval lengths of σ and σ'. Pratt first defined fusion for describing the semantics of a process logic [62] and called it *fusion product*.

It is worth comparing chop and fusion. Fusion is a general operation definable for such things as strings (i.e., sequences of letters) or intervals (i.e., sequences of states). As used here, it starts with two suitable intervals and joins them together. In contrast, chop is a logical operator which starts with an overall interval and then tests for the existence of a way to split it into two fusible subintervals. Furthermore, the semantics of the chop operator can be defined using fusion, whereas fusion is for our purposes a semantic concept, not a logical construct.

Here is a lemma relating chop with fusion:

Lemma 28 *A* PITL *formula $A;B$ is satisfiable iff there exist two intervals σ and σ' such that the fusion of them $\sigma \circ \sigma'$ is defined and one of the following is true:*

- *The interval σ has finite length, it satisfies A and the interval σ' satisfies B.*
- *The interval σ has infinite length and it satisfies A.*

This lemma provides a way to reduce the problem of constructing an interval satisfying $A;B$ to that of constructing intervals satisfying A and B.

Before further reducing transition configurations involving infinite time, we introduce the notion of *canonical intervals* and discuss their use in relating the satisfiability of chop and chop-omega formulas with satisfiability of their operands.

The next definition of a notion of canonical states and intervals together with the subsequent Lemma 30 will be extensively utilised to facilitate reasoning about intervals.

Definition 29 (Canonical States and Intervals) *For any finite set of variables V and state s, we say that s is a V-*state *if s assigns each variable* not *in V the value false.*

*Similarly, for any finite set of variables V and interval σ, we say that σ is a V-*interval *if σ's states all assign each variable* not *in V the value false.*

Furthermore, for any set of variables V, we can denote a finite V-state by the unique V-atom which the state satisfies. In addition, a V-interval can be denoted the unique sequence of V-atoms associated with its V-states.

For example, for any V-atoms α and β, the two-atom sequence $\alpha\beta$ denotes a finite V-interval with V-states denoted by α and β, respectively. Hence, $\alpha\beta \models X$ denotes that the two-state V-interval $\alpha\beta$ satisfies the formula X. If X is in PTL_V, then $\alpha\beta \models X$ holds iff the conjunction $\alpha \wedge \bigcirc\beta? \wedge X$ is satisfiable. Furthermore a single V-atom can be regarded as a one-state V-interval. For example, $\alpha \models X$ denotes that the one-state V-interval α satisfies X. For any X in PTL_V, this is the case iff the conjunction $\alpha \wedge X \wedge \mathit{empty}$ is satisfiable.

Similarly, the notation $\alpha\beta\alpha \models X$ denotes that the V-interval $\alpha\beta\alpha$, which has two identical states, satisfies the formula X.

The next lemma ensures that any satisfiable PITL_V formula is satisfied by some V-interval.

Lemma 30 *An interval σ satisfies a PITL_V formula A iff there exists a V-interval with the same number of states as σ, agrees with σ on the values of the variables in V and moreover satisfies A.*

PROOF Let σ' be the V-interval obtained from σ by setting all variables not in the set V to *false* in each state. The semantics in PITL of A ignores such variables. □

The following lemma employs V-atoms and the PTL construct *finite* to express a simple sufficient condition which ensures that any two intervals which respectively satisfy the two parts of a chop formula with a particular syntax given in the lemma can be fused together into an interval which satisfies the overall chop formula.

Lemma 31 *For any V-atom α and PITL_V formulas A and B, the following are equivalent:*

(a) The formula $(A \wedge \mathit{finite}); (\alpha \wedge B)$ is satisfiable.

(b) The formulas $A \wedge \mathit{sfin}\ \alpha$ and $\alpha \wedge B$ are satisfiable.

PROOF *(a) ⇒ (b):* If some interval σ satisfies the formula $(A \wedge \mathit{finite}); (\alpha \wedge B)$, then by the semantics of chop there exist two subintervals of σ denoted here as σ' and σ'' such that the subinterval σ' satisfies $A \wedge \mathit{finite}$ and moreover if σ' has finite length, then σ'' satisfies $\alpha \wedge B$. The right subformula *finite* in $A \wedge \mathit{finite}$ ensures that σ' is indeed finite and therefore σ'' does satisfies $\alpha \wedge B$.

(b) ⇒ (a): If the two formulas $A \wedge \mathit{sfin}\ \alpha$ and $\alpha \wedge B$ are satisfiable, then by Lemma 30 some V-intervals σ and σ' satisfy them. Now σ is finite due to the subformula $\mathit{sfin}\ \alpha$. Also, the last state of σ and the first state of σ' both equal the V-state denoted by the V-atom α. Hence σ and σ' can be fused and the fusion $\sigma \circ \sigma'$ satisfies the formula $(A \wedge \mathit{finite}); (\alpha \wedge B)$. □

5.2.3 Periodic Models and Reduction to Finite-Time Behaviour

The remaining material in this section deals with relating transition configurations involving infinite time to other formulas involving periodicity as well as to formulas about finite time. The connections are interesting in themselves and also later utilised.

The next Lemmas 32 and 33 help to establish small models, decidability and axiomatic completeness for periodic transition configurations:

Lemma 32 *For any V-atom α and PITL_V formula A, the following are equivalent:*

(a) The formula $(\alpha \wedge A)^\omega$ is satisfiable.

(b) The formula $(\alpha \wedge A)^\omega$ has a periodic model.

(c) The formula $\alpha \wedge A \wedge \bigcirc\, \mathit{sfin}\ \alpha$ is satisfiable (in finite time).

PROOF *(a) ⇒ (c):* Suppose the interval σ satisfies $(\alpha \wedge A)^\omega$. We can assume each iteration of $\alpha \wedge A$ occurs in a nonempty, finite interval as expressed by the next valid equivalence:

$$\models \quad (\alpha \wedge A)^\omega \;\equiv\; (\alpha \wedge A \wedge \mathit{finite} \wedge \mathit{more})^\omega \quad .$$

Furthermore, each pair of adjacent iterations share a common state satisfying α and hence all have α true at the beginning and end as is captured by the following valid equivalence:

$$\models \quad (\alpha \wedge A)^\omega \;\equiv\; (\alpha \wedge A \wedge \mathit{finite} \wedge \mathit{more} \wedge \mathit{fin}\ \alpha)^\omega \quad .$$

Therefore the subformula $\alpha \wedge A \wedge \mathit{finite} \wedge \mathit{more} \wedge \mathit{fin}\ \alpha$ is satisfiable (in finite time) and hence the semantically equivalent formula $\alpha \wedge A \wedge \bigcirc\, \mathit{sfin}\ \alpha$ is also satisfiable.

(c) $\Rightarrow$ *(b):* Suppose the interval σ satisfies $\alpha \wedge A \wedge \bigcirc\, sfin\ \alpha$. As a consequence of α being a V-atom and A being a PITL$_V$ formula together with Lemma 30, we can assume without loss of generality that σ is a V-interval. We then readily fuse ω instances of σ together to obtain a periodic interval satisfying the formula $(\alpha \wedge A)^\omega$.

(b) $\Rightarrow$ *(a):* Clearly if some periodic interval satisfies $(\alpha \wedge A)^\omega$, then this formula is satisfiable. $\square$

Lemma 33 shows that any satisfiable periodic transition configuration has a periodic model. Subsequently, Theorem 36 establishes that any satisfiable infinite-time transition configuration has an ultimately periodic model (i.e., an interval with a periodic suffix):

Lemma 33 *For any V-atom α, the following are equivalent:*

(a) The periodic transition configuration $\Box T \wedge \alpha \wedge L \wedge \Box \Diamond^+(\alpha \wedge L)$ is satisfiable.

(b) The periodic transition configuration $\Box T \wedge \alpha \wedge L \wedge \Box \Diamond^+(\alpha \wedge L)$ has a periodic model.

(c) The formula $(\$T)^* \wedge \alpha \wedge L \wedge \bigcirc\, sfin\ \alpha$ is satisfiable (in finite time).

Proof Theorem 24 reduces the periodic transition configuration to the semantically equivalent PITL$_V$ formula $((\$T)^* \wedge \alpha \wedge L)^\omega$. We then utilise Lemma 32. $\square$

Lemma 34 *For any V-atom α and* PITL$_V$ *formulas A and B, the following are equivalent:*

(a) The formula $(A \wedge finite); (\alpha \wedge B)^\omega$ is satisfiable.

(b) The formula $(A \wedge finite); (\alpha \wedge B)^\omega$ has an ultimately periodic model (i.e., an interval with a periodic suffix).

(c) The formula $(A \wedge finite); (\alpha \wedge B \wedge \bigcirc\, sfin\ \alpha)$ is satisfiable (in finite time).

Proof *(a)* $\Rightarrow$ *(c):* If the formula $(A \wedge finite); (\alpha \wedge B)^\omega$ is satisfiable then the PITL$_V$ formula $(A \wedge finite); (\alpha \wedge B \wedge \bigcirc\, sfin\ \alpha)^\omega$ is also satisfiable. From this readily follows the satisfiability of the formula $(A \wedge finite); (\alpha \wedge B \wedge \bigcirc\, sfin\ \alpha)$.

(c) $\Rightarrow$ *(b):* If the formula $(A \wedge finite); (\alpha \wedge B \wedge \bigcirc\, sfin\ \alpha)$ is satisfiable then Lemma 31 ensures that the two formulas $A \wedge finite \wedge fin\ \alpha$ and $\alpha \wedge B \wedge \bigcirc\, sfin\ \alpha$ are also satisfiable. Lemma 32 then yields that the formula $(\alpha \wedge B)^\omega$ has a periodic model. Suppose the interval σ satisfies $A \wedge finite \wedge fin\ \alpha$ and the interval σ' is a periodic model of $(\alpha \wedge B)^\omega$. Lemma 30 permits us to assume that σ and σ' are V-intervals. We can fuse σ together with σ' to obtain an ultimately periodic model for $(A \wedge finite); (\alpha \wedge B)^\omega$.

(b) $\Rightarrow$ *(a):* Clearly if some ultimately periodic interval satisfies $(A \wedge finite); (\alpha \wedge B)^\omega$, then this formula is satisfiable. $\square$

Lemma 35 *For any* PITL$_V$ *formulas A and B, the following are equivalent:*

(a) The formula $(A \wedge finite); (B \wedge (\vec{V} \leftarrow \vec{V}))^\omega$ is satisfiable.

(b) The formula $(A \wedge finite); (B \wedge (\vec{V} \leftarrow \vec{V}))^\omega$ has an ultimately periodic model.

(c) The formula $(A \wedge finite); (B \wedge more \wedge finite \wedge (\vec{V} \leftarrow \vec{V}))$ is satisfiable (in finite time).

Proof This follows from Lemma 34 and simple temporal reasoning involving chop and the operator $\bigvee$. We also make use of the following valid equivalences concerning $\vec{V} \leftarrow \vec{V}$, the formula B and any V-atom α:

$$\begin{array}{ll} \models & \alpha \wedge B \wedge \bigcirc\, sfin\ \alpha \quad \equiv \quad \alpha \wedge B \wedge more \wedge finite \wedge (\vec{V} \leftarrow \vec{V}) \\ \models & (\alpha \wedge B)^\omega \quad \equiv \quad \alpha \wedge (B \wedge (\vec{V} \leftarrow \vec{V}))^\omega \ . \end{array}$$

$\square$

Theorem 36 *The following are equivalent:*

(a) The infinite-time transition configuration $\Box T \wedge init \wedge \Box \Diamond^+ L$ is satisfiable.

Type of transition configuration	Upper bounds	Where proved
Finite-time	Interval length less than $\lvert Atoms_V \rvert$	Theorem 38
Infinite-time	Initial part $< \lvert Atoms_V \rvert$, Period $\leq (\lvert L \rvert + 1) \cdot \lvert Atoms_V \rvert$	Theorem 45
Final	Interval length is 0	straightforward
Periodic	Period $\leq (\lvert L \rvert + 1) \cdot \lvert Atoms_V \rvert$	Lemma 44

Table 4: Summary of upper bounds of intervals for transition configurations

(b) The infinite-time transition configuration $\Box T \wedge init \wedge \Box \Diamond^+ L$ has an ultimately periodic model.

(c) The PITL$_V$ *formula $\big((\$T)^* \wedge init \wedge finite\big); \big((\$T)^* \wedge L \wedge more \wedge finite \wedge (\vec{V} \leftarrow \vec{V})\big)$ is satisfiable (in finite time).*

(d) The PTL$_V$ *formula $\boxdot T \wedge init \wedge \Diamond(L \wedge finite \wedge more \wedge (\vec{V} \leftarrow \vec{V}))$ is satisfiable (in finite time).*

Proof We need to obtain formulas which are in a form suitable for Lemma 35. First of all, Theorem 26 permits us to re-express the infinite-time transition configuration $\Box T \wedge init \wedge \Box \Diamond^+ L$ as the formula $((\$T)^* \wedge init \wedge finite); \big((\$T)^* \wedge L \wedge (\vec{V} \leftarrow \vec{V})\big)^{\omega}$. Recall that Theorem 11 shows the semantic equivalence of the formulas $\boxdot T$ and $(\$T)^*$. Therefore, simple interval-based temporal reasoning ensures that formulas in (c) and (d) are semantically equivalent. We complete the proof by invoking Lemma 35. □

6 Small Models for Transition Configurations

We now turn to giving upper bounds on small models for satisfiable transition configurations. This is later used in Sect. 8 to construct a decision procedure for them. Table 4 summarises the upper bounds for intervals satisfying the various kinds of transition configurations and where the results are proved.

It will be necessary to employ the fact (e.g., in Theorem 38 and Lemma 42) that the formula $\alpha \wedge (\$T)^* \wedge sfin\ \beta$ is satisfiable iff a simple variant of it is satisfiable in an interval of bounded interval length. The following lemma deals with this:

Lemma 37 *For any V-atoms α and β, the formula $\alpha \wedge (\$T)^* \wedge sfin\ \beta$ is satisfiable iff the formula $\alpha \wedge (\$T)^{<\lvert Atoms_V \rvert} \wedge sfin\ \beta$ is satisfiable. Hence, the formula $\alpha \wedge (\$T)^* \wedge sfin\ \beta$ is satisfiable iff it is satisfiable in an interval having interval length less than $\lvert Atoms_V \rvert$.*

Proof Any interval satisfying $\alpha \wedge (\$T)^{<\lvert Atoms_V \rvert} \wedge sfin\ \beta$ can be readily seen to also satisfy $\alpha \wedge (\$T)^* \wedge sfin\ \beta$. Let us now establish the converse by doing a proof by contradiction. Suppose $\alpha \wedge (\$T)^* \wedge sfin\ \beta$ is satisfiable but $\alpha \wedge (\$T)^{<\lvert Atoms_V \rvert} \wedge sfin\ \beta$ is not. Let σ be any interval which has the smallest length of those which satisfy $\alpha \wedge (\$T)^* \wedge sfin\ \beta$. Lemma 30 permits us to assume that σ is a V-interval. Now σ's length is greater than or equal to $\lvert Atoms_V \rvert$ and therefore contains at least $\lvert Atoms_V \rvert + 1$ states. Consequently, some V-state occurs at least twice in σ. Let the V-atom γ denote this state. It follows that σ satisfies the following PITL$_V$ formula:

$$\alpha \wedge \big((\$T)^*; \gamma?; (\$T)^+; \gamma?; (\$T)^*\big) \wedge sfin\ \beta \ .$$

Therefore σ contains two proper subintervals σ' and σ'' which respectively satisfy the PITL$_V$ formulas $\alpha \wedge (\$T)^* \wedge sfin\ \gamma$ and $\gamma \wedge (\$T)^* \wedge sfin\ \beta$. In addition, the last state of σ' is the same as the first one of σ'' so σ' and σ'' can be fused together. The fusion $\sigma' \circ \sigma''$ has length strictly less than that of σ and furthermore, like σ, satisfies the formula $\alpha \wedge (\$T)^* \wedge sfin\ \beta$. But this violates the assumption that σ was amongst the shortest such intervals and yields a contradiction. □

Theorem 38 *If a finite-time transition configuration $\Box T \wedge init \wedge finite$ is satisfiable, then it is satisfied by some finite interval of length less than $\lvert Atoms_V \rvert$.*

PROOF Theorem 17 ensures that the finite-time transition configuration $\Box T \wedge \mathit{init} \wedge \mathit{finite}$ is semantically equivalent to the formula $((\$T)^* \wedge \mathit{init} \wedge \mathit{finite}); (T \wedge \mathit{empty})$. This is satisfiable iff for some V-atom α, the formula $((\$T)^* \wedge \mathit{init} \wedge \mathit{finite}); (\alpha \wedge T \wedge \mathit{empty})$ is satisfiable. Now Lemma 31 ensures that this itself is satisfiable iff the formulas $(\$T)^* \wedge \mathit{init} \wedge \mathit{sfin}\ \alpha$ and $\alpha \wedge T \wedge \mathit{empty}$ are both satisfiable. By Lemma 37, the first of these is satisfiable iff the formula $(\$T)^{<|Atoms_V|} \wedge \mathit{init} \wedge \mathit{sfin}\ \alpha$ is satisfiable. Lemma 30 permits us to assume without loss of generality that the intervals satisfying the formulas $(\$T)^{<|Atoms_V|} \wedge \mathit{init} \wedge \mathit{sfin}\ \alpha$ and $\alpha \wedge T \wedge \mathit{empty}$ are V-intervals. We then fuse the intervals together to obtain one of interval length less than $|Atoms_V|$ which satisfies $((\$T)^* \wedge \mathit{init} \wedge \mathit{finite}); (T \wedge \mathit{empty})$ and hence also satisfies the semantically equivalent finite-time transition configuration. □

The next definition is required for analysing infinite-time configurations and makes use of the earlier Definitions 4–6 concerning conjunctions and Definition 8 concerning conditional liveness formulas

Definition 39 (Enabled Liveness Formula) *An* enabled liveness formula *En is a conjunction of $|En|$ formulas in which for each $k : 1 \leq k \leq |En|$, the subformula $En[k]$ is of the form $\diamond\!\!\!\!\cdot\ w$, for some state formula w. The state formulas $\theta_{En[1]}, \ldots, \theta_{En[|En|]}$ denote the $|En|$ liveness tests in En so that $En[k]$ and $\diamond\!\!\!\!\cdot\ \theta_{En[k]}$ refer to the same formula.*

For any V-atom α and conditional liveness formula L, we will also define $En_{L,\alpha}$ to be the enabled liveness formula containing the L's liveness tests which are enabled by α (recall Definition 8). Let S be the set of indices of L's implications which are enabled by α. Then $En_{L,\alpha}$ is the conjunction $\bigwedge_{j \in S} \diamond\!\!\!\!\cdot\ \theta_{L[j]}$.

For example, suppose V is the set $\{p, q\}$, α is the V-atom $\neg p \wedge q$ and L is the conditional liveness formula $((p \vee \neg q) \supset \diamond\!\!\!\!\cdot\ \neg p) \wedge (q \supset \diamond\!\!\!\!\cdot(p \equiv \neg q)) \wedge (\mathit{true} \supset \diamond\!\!\!\!\cdot(p \supset q))$ mentioned earlier as formula (1). Then $En_{L,\alpha}$ is the conjunction $\diamond\!\!\!\!\cdot(p \equiv \neg q) \wedge \diamond\!\!\!\!\cdot(p \supset q)$.

Lemma 40 *For any V-atom α and conditional liveness formula L in PTL_V, the conjunctions $\alpha \wedge L$ and $\alpha \wedge En_{L,\alpha}$ are semantically equivalent*

Not surprisingly, the hardest part of the proof of existence of small models for infinite-time transition configurations involves finding small models for periodic transition configurations. Recall that Lemma 33 relates the satisfiability of the periodic transition configuration $\Box T \wedge \alpha \wedge L \wedge \Box \Diamond^+(\alpha \wedge L)$ to that of the PITL_V formula $(\$T)^* \wedge \alpha \wedge L \wedge \bigcirc\, \mathit{sfin}\ \alpha$. We will use the equivalence of $\alpha \wedge L$ and $\alpha \wedge En_{L,\alpha}$ to assist in the analysis of bounded models of $(\$T)^* \wedge \alpha \wedge L \wedge \bigcirc\, \mathit{sfin}\ \alpha$. These can then be used to obtain a bounded periodic model for the original periodic transition configuration.

Lemma 41 *For any V-atom α and conditional liveness formula L in PTL_V, the following equivalence is valid:*

$$\models \quad (\$T)^* \wedge \alpha \wedge L \wedge \bigcirc\, \mathit{sfin}\ \alpha \quad \equiv \quad (\$T)^* \wedge \alpha \wedge En_{L,\alpha} \wedge \bigcirc\, \mathit{sfin}\ \alpha \ .$$

PROOF This readily follows from the earlier Lemma 40 concerning the semantic equivalence of the formulas $\alpha \wedge L$ and $\alpha \wedge En_{L,\alpha}$. □

The next Lemma 42 shortens the nonempty, finite model expressed by the formula $(\$T)^* \wedge \alpha \wedge En \wedge \bigcirc\, \mathit{sfin}\ \alpha$ to one having a bounded length by adapting the technique presented earlier in Lemma 37 concerning a bounded model for the formula $(\$T)^* \wedge \alpha \wedge \mathit{sfin}\ \beta$.

Lemma 42 *For any V-atom α and enabled liveness formula En in PTL_V, if the formula $(\$T)^* \wedge \alpha \wedge En \wedge \bigcirc\, \mathit{sfin}\ \alpha$ is satisfiable, then it is satisfied by a interval having interval length at most $(|En| + 1)\, |Atoms_V|$.*

PROOF If the formula $(\$T)^* \wedge \alpha \wedge En \wedge \bigcirc\, \mathit{sfin}\ \alpha$ is satisfiable, then by Lemma 30 there exists some satisfying V-interval. We can fuse $|En| + 1$ copies of this interval together to obtain a V-interval σ which satisfies the formula $\big((\$T)^* \wedge \alpha \wedge En \wedge \mathit{finite}\big)^{|En|+1} \wedge \bigcirc\, \mathit{sfin}\ \alpha$. It is not hard to check than σ itself satisfies the original formula $(\$T)^* \wedge \alpha \wedge En \wedge \bigcirc\, \mathit{sfin}\ \alpha$ since each liveness test in En is satisfied somewhere in σ prior to the last state. Furthermore, there exist a sequence of $|En|$ V-atoms $\gamma_1, \ldots, \gamma_{|En|}$ such that for each $j : 1 \leq j \leq |En|$, the state formula $\gamma_j \wedge \theta_{En[j]}$ is satisfied by some state prior to the last one and the V-interval σ satisfies the next formula:

$$\alpha \;\wedge\; \big((\$T)^*;\gamma_1?;\ldots;(\$T)^*;\gamma_{|En|}?;(\$T)^+\big) \;\wedge\; \bigcirc\, \mathit{sfin}\ \alpha\ .$$

If a gap between two of the $|En|$ selected states satisfying their respective liveness tests has interval length of at least $|Atoms_V|$, then within the gap, some state occurs twice. Such a gap can then be shortened in the manner of Lemma 37. By means of this we obtain from the V-interval σ another V-interval having bounded length and satisfying the formula below:

$$\begin{aligned}&\alpha \;\wedge\; \big((\$T)^{<|Atoms_V|};\gamma_1?;\ldots;(\$T)^{<|Atoms_V|};\gamma_{|En|}?;(\$T)^{\leq|Atoms_V|}\big)\\ &\quad\wedge\; \bigcirc\, \mathit{sfin}\ \alpha\ .\end{aligned}$$

The resulting new interval is nonempty and has interval length not exceeding $(|En|+1)\,|Atoms_V|$. Moreover it still satisfies $(\$T)^* \wedge \alpha \wedge En \wedge \bigcirc\, \mathit{sfin}\ \alpha$. □

Lemma 43 *If the formula $(\$T)^* \wedge \alpha \wedge L \wedge \bigcirc\, \mathit{sfin}\ \alpha$ is satisfiable, then it is satisfiable on a finite, nonempty interval with interval length at most $(|L|+1)\,|Atoms_V|$.*

Proof From Lemma 42 we have that if the formula $(\$T)^* \wedge \alpha \wedge En_{L,\alpha} \wedge \bigcirc\, \mathit{sfin}\ \alpha$ is satisfiable, then it is satisfiable on a finite, nonempty interval having interval length at most $(|En_{L,\alpha}|+1)\,|Atoms_V|$. Lemma 40 ensures that the conjunctions $\alpha \wedge L$ and $\alpha \wedge En_{L,\alpha}$ are semantically equivalent. In addition, we have $|En_{L,\alpha}| \leq |L|$. Therefore, if the formula $(\$T)^* \wedge \alpha \wedge L \wedge \bigcirc\, \mathit{sfin}\ \alpha$ is satisfiable, then it is satisfiable on a finite, nonempty interval with interval length at most $(|L|+1)\,|Atoms_V|$. □

Lemma 44 *If the periodic transition configuration $\Box T \wedge \alpha \wedge L \wedge \Box\Diamond^+(\alpha \wedge L)$ is satisfiable, then it is satisfied by a periodic interval with period of interval length at most $(|L|+1)\,|Atoms_V|$.*

Proof Lemma 33 ensures that if the periodic transition configuration is satisfiable, then the formula $(\$T)^* \wedge \alpha \wedge L \wedge \bigcirc\, \mathit{sfin}\ \alpha$ is satisfiable. By Lemma 43, if this is satisfiable, then it has a satisfying interval having interval length at most $(|L|+1)\,|Atoms_V|$. Lemma 30 permits us to assume without loss of generality that the interval is a V-interval. We can then fuse ω copies of it together to obtain a periodic interval which has a period with interval length at most $(|L|+1)\,|Atoms_V|$ and also satisfies the formula $((\$T)^* \wedge \alpha \wedge L)^\omega$. Theorem 24 establishes that this formula is equivalent to the original periodic transition configuration. □

Theorem 45 *If the infinite-time transition configuration $\Box T \wedge \mathit{init} \wedge \Box\Diamond^+ L$ is satisfiable, then it is satisfied by an ultimately periodic interval consisting of an initial segment having interval length less than $|Atoms_V|$ fused with a periodic interval having a period with interval length of at most $(|L|+1)\,|Atoms_V|$.*

Proof If some interval satisfies the formula $\Box T \wedge \mathit{init} \wedge \Box\Diamond^+ L$, then Lemma 18 ensures that the interval also satisfies the next semantically equivalent formula:

$$((\$T)^* \wedge \mathit{init} \wedge \mathit{finite});\bigvee_{\alpha\in Atoms_V}(\Box T \wedge \alpha \wedge L \wedge \Box\Diamond^+(\alpha \wedge L))\ . \tag{10}$$

Lemma 31 and simple temporal reasoning establish that for some V-atom α the two formulas $(\$T)^* \wedge \mathit{init} \wedge \mathit{sfin}\ \alpha$ and $\Box T \wedge \alpha \wedge L \wedge \Box\Diamond^+(\alpha \wedge L)$ are satisfiable. By Lemma 37, the first formula is satisfiable in some interval σ having interval length less than $|Atoms_V|$. Lemma 44 yields some periodic interval σ' which satisfies the second formula and possesses a period with interval length of at most $(|L|+1)\,|Atoms_V|$. Lemma 30 permits us to assume that σ and σ' are V-intervals. Therefore the last state of σ is the same as the first one of σ' since both states satisfy α. The fusion $\sigma \circ \sigma'$ is itself ultimately periodic and satisfies the formula (10). Hence it also satisfies the semantically equivalent original infinite-time transition configuration $\Box T \wedge \mathit{init} \wedge \Box\Diamond^+ L$ as well. In addition, the interval $\sigma \circ \sigma'$ has an initial segment having interval length less than $|Atoms_V|$ fused with a periodic interval with period of interval length at most $(|L|+1)\,|Atoms_V|$. □

7 Decomposition of Transition Configurations

We now prove the two Theorems 46 and 49 which respectively relate the satisfiability of finite-time and infinite-time transition configurations with simple interval-oriented tests involving finite time. These theorems are later used in Sect. 8 as part of the justification of the our PTL decision procedure and in Sect. 10 as part of the completeness proof of an axiom system for PTL.

Theorem 46 (Decomposing Finite-Time Transition Configurations) *The following are equivalent:*

(a) The finite-time configuration $\Box T \wedge \mathit{init} \wedge \mathit{finite}$ is satisfiable.

(b) For some V-atoms α and β, the three formulas below are satisfiable:

$$\alpha \wedge \mathit{init} \qquad (\$T)^* \wedge \alpha \wedge \mathit{sfin}\ \beta \qquad T \wedge \beta \wedge \mathit{empty}\ .$$

Proof Theorem 17 ensures that the finite-time configuration is semantically equivalent to the next PITL$_V$ formula:

$$((\$T)^* \wedge \mathit{init} \wedge \mathit{finite}); (T \wedge \mathit{empty})\ .$$

Now simple interval-based reasoning guarantees that this is satisfiable iff for some V-atoms α and β, the next formula is satisfiable:

$$((\$T)^* \wedge \alpha \wedge \mathit{init} \wedge \mathit{finite}); (T \wedge \beta \wedge \mathit{empty})\ .$$

Lemma 31 ensures that this is itself satisfiable iff the next two formulas are:

$$(\$T)^* \wedge \alpha \wedge \mathit{init} \wedge \mathit{sfin}\ \beta \qquad T \wedge \beta \wedge \mathit{empty}\ .$$

Finally, simple temporal reasoning ensures that the first of these is itself is satisfiable iff the following two formulas are satisfiable:

$$\alpha \wedge \mathit{init} \qquad (\$T)^* \wedge \alpha \wedge \mathit{sfin}\ \beta\ .$$

□

We now turn to decomposing an infinite-time transition configuration:

Lemma 47 *The infinite-time transition configuration $\Box T \wedge \mathit{init} \wedge \Box \Diamond^+ L$ is satisfiable iff for some V-atoms α and β, the following formulas are satisfiable:*

$$(\$T)^* \wedge \alpha \wedge \mathit{init} \wedge \mathit{sfin}\ \beta \qquad (\$T)^* \wedge \beta \wedge \mathit{En}_{L,\beta} \wedge \bigcirc\, \mathit{sfin}\ \beta\ . \tag{11}$$

Proof Theorem 36 ensures that the infinite-time configuration is satisfiable iff the next PITL$_V$ formula is satisfiable:

$$\big((\$T)^* \wedge \mathit{init} \wedge \mathit{finite}\big); \big((\$T)^* \wedge L \wedge \mathit{more} \wedge \mathit{finite} \wedge (\vec{V} \leftarrow \vec{V})\big)\ .$$

Simple interval-based temporal reasoning ensures that this itself is satisfiable iff for some V-atoms α and β, next formula is satisfiable:

$$\big((\$T)^* \wedge \alpha \wedge \mathit{init} \wedge \mathit{finite}\big); \big((\$T)^* \wedge \beta \wedge L \wedge \bigcirc\, \mathit{sfin}\ \beta\big)\ . \tag{12}$$

Now Lemma 40 guarantees the semantic equivalence of the conjunctions $\beta \wedge L$ and $\beta \wedge \mathit{En}_{L,\beta}$. We therefore can replace L by $\mathit{En}_{L,\beta}$ in formula (12). Finally, Lemma 31 yields that the resulting formula is itself satisfiable iff the two formulas in (11) are satisfiable. □

The next lemma concerning enabled liveness formulas is shortly used in Theorem 49 to analyse the satisfiability of infinite-time configurations:

Lemma 48 *For any V-atom α and enabled liveness formula En, the following are equivalent:*

(a) The formula $(\$T)^* \wedge \alpha \wedge \mathit{En} \wedge \bigcirc\, \mathit{sfin}\ \alpha$ is satisfiable.

(b) For some $|En|$ V-atoms $\gamma_1, \ldots, \gamma_{|En|}$ (not necessarily distinct), the following are all satisfiable:

$$(\$T)^* \wedge \alpha \wedge \bigcirc\, \mathit{sfin}\ \alpha$$
$$\text{for each } \gamma_i\colon\quad (\$T)^* \wedge \alpha \wedge \mathit{sfin}\ \gamma_i \quad \gamma_i \wedge \theta_{En_{L,\alpha}[i]} \quad (\$T)^* \wedge \gamma_i \wedge \mathit{sfin}\ \alpha\ .$$

Proof Induction on the length of En and simple interval-based reasoning can be used to demonstrate that the formula $(\$T)^* \wedge \alpha \wedge En \wedge \bigcirc\, \mathit{sfin}\ \alpha$ is satisfiable iff the formula $(\$T)^* \wedge \alpha \wedge \bigcirc\, \mathit{sfin}\ \alpha$ is satisfiable and also for some V-atoms $\gamma_1, \ldots, \gamma_{|En|}$, for each γ_i the following formula is satisfiable:

$$(\$T)^* \wedge \alpha \wedge \Diamond(\gamma_i \wedge \theta_{En[i]}) \wedge \mathit{sfin}\ \alpha\ . \tag{13}$$

This guarantees that for each liveness test $\theta_{En[i]}$ in En, the V-atom α can reach some V-atom γ_i which satisfies $\theta_{En[i]}$ and this V-atom γ_i itself can reach back to α. We can re-express (13) as the semantically equivalent formula below:

$$\big((\$T)^* \wedge \alpha \wedge \mathit{finite}\big); \big((\$T)^* \wedge \gamma_i \wedge \theta_{En[i]} \wedge \mathit{sfin}\ \alpha\big)\ .$$

Lemma 31 ensures that this is satisfiable iff the next two formulas are:

$$(\$T)^* \wedge \alpha \wedge \mathit{sfin}\ \gamma_i \qquad (\$T)^* \wedge \gamma_i \wedge \theta_{En[i]} \wedge \mathit{sfin}\ \alpha\ .$$

The second one is satisfiable iff the two formulas shown below are satisfiable:

$$\gamma_i \wedge \theta_{En[i]} \qquad (\$T)^* \wedge \gamma_i \wedge \mathit{sfin}\ \alpha\ .$$

□

Theorem 49 (Decomposing Infinite-Time Transition Configurations) *The following are equivalent:*

(a) The infinite-time configuration $\Box T \wedge \mathit{init} \wedge \Box \Diamond^+ L$ is satisfiable.

(b) For some V-atoms α, β and $\gamma_1, \ldots, \gamma_{|En_{L,\beta}|}$ (not necessarily distinct), the following are all satisfiable:

$$\alpha \wedge \mathit{init} \qquad (\$T)^* \wedge \alpha \wedge \mathit{sfin}\ \beta \qquad (\$T)^* \wedge \beta \wedge \bigcirc\, \mathit{sfin}\ \beta$$
$$\text{for each } \gamma_i\colon\quad (\$T)^* \wedge \beta \wedge \mathit{sfin}\ \gamma_i \quad \gamma_i \wedge \theta_{En_{L,\beta}[i]} \quad (\$T)^* \wedge \gamma_i \wedge \mathit{sfin}\ \beta\ .$$

Proof Lemma 47 establishes that the infinite-time configuration $\Box T \wedge \mathit{init} \wedge \Box \Diamond^+ L$ is satisfiable iff there exist some V-atoms α and β for which the next two formulas are satisfiable:

$$(\$T)^* \wedge \alpha \wedge \mathit{init} \wedge \mathit{sfin}\ \beta \qquad (\$T)^* \wedge \beta \wedge En_{L,\beta} \wedge \bigcirc\, \mathit{sfin}\ \beta\ . \tag{14}$$

Now simple temporal reasoning ensures that the first of these is itself is satisfiable iff the following two formulas are satisfiable:

$$\alpha \wedge \mathit{init} \qquad (\$T)^* \wedge \alpha \wedge \mathit{sfin}\ \beta\ .$$

Furthermore, Lemma 48 guarantees that the second formula in (14) is satisfiable iff the formula $(\$T)^* \wedge \beta \wedge \bigcirc\, \mathit{sfin}\ \beta$ is satisfiable and furthermore for some V-atoms $\gamma_1, \ldots, \gamma_{|En_{L,\beta}|}$ (not necessarily distinct), the following are all satisfiable for each γ_i:

$$(\$T)^* \wedge \beta \wedge \mathit{sfin}\ \gamma_i \qquad \gamma_i \wedge \theta_{En_{L,\beta}[i]} \qquad (\$T)^* \wedge \gamma_i \wedge \mathit{sfin}\ \beta\ .$$

□

8 A Decision Procedure

We now describe a decision procedure for finite-time and infinite-time transition configurations based on Binary Decision Diagrams (BDDs) [8, 9] which provide an efficient basis for performing many computational tasks involving reductions to reasoning about formulas in propositional logic. We had little difficulty implementing the decision procedure using the popular Colorado University Decision Diagram Package

(CUDD) [19] developed by Somenzi. Our prototype tool consists of a front-end coded in the CLISP [15] implementation of Common Lisp [1] as well as a back-end coded in Perl [59]. The back-end employs a Perl-oriented interface to CUDD written by Somenzi and called PerlDD [60]. The front-end accepts arbitrary PTL formulas and converts them to transition configurations using methods later described in Sections 11 and 12. The transition configurations are then passed to the back-end which analyses them using BDDs. In this section we describe the basis for performing this analysis.

The remainder of this section assumes that the reader already has some familiarity with BDDs.

Our algorithm for finite-time transition configurations adapts methods for *symbolic state space traversal* described by Coudert, Berthet and Madre [16–18] (see also Kropf [14, 41]) for use with BDD-based representations of formulas in propositional logic. It simultaneously greatly benefits from closely related methods first employed by McMillan in symbolic model checking [10, 14, 46] which also include the automatic generation of counterexamples for unsatisfiable formulas and, similarly, witnesses for satisfiable ones. Recall that Theorem 46 shows that the finite-time transition configuration $\Box T \wedge init \wedge finite$ is satisfiable iff for some V-atoms α and β, the next three formulas are satisfiable:

$$\alpha \wedge init \qquad (\$T)^* \wedge \alpha \wedge sfin\ \beta \qquad T \wedge \beta \wedge empty\ .$$

We can readily search for suitable V-atoms using BDDs. Three BDDs Γ_1, Γ_2 and Γ_3 are initially constructed. In what follows, please recall the notion $\dashv X$ introduced in Definition 1 to denote that the formula X is satisfiable. We first describe the roles of the BDDs Γ_1, Γ_2 and Γ_3 before actually constructing them:

- The BDD Γ_1 represents the state formula *init* and hence the set of V-atoms satisfying *init* (i.e., the set $\{\alpha \in Atoms_V : \alpha \models init\}$). This is the same as the set $\{\alpha \in Atoms_V : \dashv \alpha \wedge init\}$.
- The second BDD Γ_2 captures all pairs of V-atoms corresponding to unit (i.e., two-state) intervals satisfying T. In other words, it corresponds to the set $\{\langle \alpha, \beta \rangle \in Atoms_V^2 : \alpha\beta \models T\}$. This is the same as the set $\{\langle \alpha, \beta \rangle \in Atoms_V^2 : \dashv T \wedge \alpha \wedge skip \wedge sfin\ \beta\}$.
- The third BDD Γ_3 captures the behaviour of T in an empty interval. Therefore Γ_3 represents the set of all V-atoms satisfying the formula $T \wedge empty$ (i.e., the set $\{\alpha \in Atoms_V : \alpha \models T\}$). This is the same as the set $\{\alpha \in Atoms_V : \dashv T \wedge \alpha \wedge empty\}$

In the course of manipulating the BDDs we make use of two finite sets of propositional variables. They include the original ones (e.g., p, $r_1, \ldots, r_4$) as well as primed versions (e.g., p', $r'_1, \ldots, r'_4$). For convenience, we often do not distinguish between a BDD and the propositional logic formula it represents.

Let V and V' respectively denote the two sets of variables. We now construct the BDDs Γ_1, Γ_3 and Γ_2 as follows:

- Let Γ_1 be the formula *init*.
- Obtain Γ_2 from the formula T by replacing all variables in the scope of any $\bigcirc$ constructs by corresponding ones in V' and then deleting all $\bigcirc$ operators (but not the associated operands) to obtain a formula in conventional propositional logic. We refer to this process of constructing Γ_2 from T by the term *flattening*.
- Obtain Γ_3 from the formula T by replacing each $\bigcirc$ construct by *false*.

The BDDs Γ_1 and Γ_3 both only can contain variables in V whereas Γ_2 can contain variables in V and V'.

Suppose T and *init* are the following formulas mentioned earlier in §3.3:

$$\begin{array}{ll} T\colon\ (r_1 \equiv (p \vee \bigcirc r_1)) \wedge (r_2 \equiv (\neg r_1 \vee \bigcirc r_2)) & \qquad init\colon\ \neg r_2 \wedge \neg r_4\ . \\ \quad \wedge\, (r_3 \equiv (\neg p \vee \bigcirc r_3)) \wedge (r_4 \equiv (\neg r_3 \vee \bigcirc r_4)) & \end{array}$$

Here are the associated Γ_1, Γ_2 and Γ_3 for these T and *init*:

$$\begin{array}{ll} \Gamma_1\colon & \neg r_2 \wedge \neg r_4 \\ \Gamma_2\colon & (r_1 \equiv (p \vee r'_1)) \wedge (r_2 \equiv (\neg r_1 \vee r'_2)) \\ & \wedge\, (r_3 \equiv (\neg p \vee r'_3)) \wedge (r_4 \equiv (\neg r_3 \vee r'_4)) \\ \Gamma_3\colon & (r_1 \equiv (p \vee false)) \wedge (r_2 \equiv (\neg r_1 \vee false)) \\ & \wedge\, (r_3 \equiv (\neg p \vee false)) \wedge (r_4 \equiv (\neg r_3 \vee false))\ . \end{array}$$

The connection between the BDDs for Γ_1 and Γ_3 and the previously mentioned sets of V-atoms they are meant to capture is straightforward. In order to justify the less intuitive relationship between the construction for Γ_2 and the earlier associated set of pairs of V-atoms, we shortly present Lemma 51 relating Γ_2 with T. However, the following lemma concerning NL^1 formulas is first given since it is used in the proof of Lemma 51.

Lemma 50 *The following are equivalent for any* NL^1 *formula* T*:*

(a) The formula T *is satisfiable in some nonempty interval.*

(b) The formula $skip \wedge T$ *is satisfiable.*

PROOF *(a)* $\Rightarrow$ *(b):* Suppose some nonempty interval σ satisfies the formula T. Now σ contains at least two states. Let σ' denote the subinterval consisting the first two states in σ. Now σ' satisfies the formula *skip*. Furthermore, the formula T is in NL^1. Lemma 10 consequently ensures that the interval σ', like σ, satisfies the formula T because both two intervals share the same first two states. Therefore σ' satisfies the formula $skip \wedge T$.

(b) $\Rightarrow$ *(a):* If some interval σ satisfies the PTL formula $skip \wedge T$, then σ is clearly nonempty and also satisfies T. □

Lemma 51 *For any* V*-atoms* α *and* β*, the following are equivalent:*

(a) The formula $T \wedge \alpha \wedge skip \wedge sfin\ \beta$ *is satisfiable (i.e.,* $\alpha\beta \models T$*).*

(b) The propositional logic formula $\Gamma_2 \wedge \alpha \wedge \beta_V^{V'}$ *is satisfiable.*

PROOF *(a)* $\Rightarrow$ *(b):* Suppose the formula $T \wedge \alpha \wedge skip \wedge sfin\ \beta$ is satisfiable. Then the flattening of T into Γ_2 readily yields that the formula $\Gamma_2 \wedge \alpha \wedge \beta_V^{V'}$ is satisfiable.

(b) $\Rightarrow$ *(a):* If the propositional logic formula $\Gamma_2 \wedge \alpha \wedge \beta_V^{V'}$ is satisfiable, then the flattening of $\bigcirc$ constructs in Γ_2 readily yields that the NL^1 formula $T \wedge \alpha \wedge \bigcirc\beta$ is satisfiable. Clearly any interval satisfying it has at least two states. Hence by the previous Lemma 50 the formula $skip \wedge T \wedge \alpha \wedge \bigcirc\beta$ is satisfiable. Simple temporal reasoning then ensures that the semantically equivalent formula $T \wedge \alpha \wedge skip \wedge sfin\ \beta$ is also satisfiable. □

We use Γ_2 together with the first BDD Γ_1 to iteratively calculate a sequence of BDDs $\Delta_0, \ldots, \Delta_k, \ldots$ so that for any k, Δ_k describes all V-atoms which can be reached from one which satisfies *init* in exactly k steps. In other words, Δ_k represents the following set:

$$\{\beta \in Atoms_V : \text{for some } \alpha \in Atoms_V,\ \dashv (\$\,T)^k \wedge \alpha \wedge init \wedge sfin\ \beta\}\ .$$

We set Δ_0 to be Γ_1. Therefore, every variable in Δ_0 is in V. Each Δ_{k+1} is calculated to be semantically equivalent to the next quantified propositional logic formula in which renaming ensures that all free variables are in V:

$$\big(\exists V.\,(\Delta_k \wedge \Gamma_2)\big)_{V'}^{V}\ . \tag{15}$$

Due to the final renaming, the sole variables left in the BDD Δ_{k+1} itself are elements of V. The only BDD operations required to calculate Δ_{k+1} from (15) are logical-and, existential quantification (which actually yields a BDD representing a semantically equivalent quantifier-free formula) and renaming which are all standard ones.

Remark 52 *Within the CUDD system, the entire calculation for obtaining* $\exists V.\,(\Delta_k \wedge \Gamma_2)$ *can even be done by a single CUDD operation tailored to handle this specific kind of common BDD manipulation. Furthermore, the renaming of variables in* V' *to those in* V *is actually achieved by taking the BDD obtained for* $\exists V.\,(\Delta_k \wedge \Gamma_2)$ *and then performing a single CUDD operation which yields another BDD in which the variables in* V *are swapped with the corresponding ones in* V'*.*

For any given Δ_k which has been calculated, we next determine the logical-and of Γ_3 and Δ_k and then proceed as follows:

1. If the logical-and is not false, then there is some V-atom β satisfying $T \wedge empty$ which can be reached in k steps from a V-atom α satisfying $init$. Therefore the next three formulas are all satisfiable:

$$\alpha \wedge init \qquad (\$T)^k \wedge \alpha \wedge sfin\ \beta \qquad T \wedge \beta \wedge empty \ .$$

   Now the second formula ensures the satisfiability of the formula $(\$T)^* \wedge \alpha \wedge sfin\ \beta$. Therefore Theorem 46 can be invoked to obtain the satisfiability of the original finite-time transition configuration $\Box T \wedge init \wedge finite$. We therefore do not need to calculate any further Δ_k's.

2. Otherwise, the logical-and is false so we must continue to iterate.

During the iteration process, we maintain a BDD representing the set of all V-atoms so far reachable from one satisfying $init$. This BDD corresponds to the formula $\bigvee_{0 \le i \le k} \Delta_i$ which equals the next set:

$$\{\beta \in Atoms_V : \text{for some } \alpha \in Atoms_V, \dashv (\$T)^{\le k} \wedge \alpha \wedge init \wedge sfin\ \beta\} \ .$$

If no such β exists which also satisfies $T \wedge empty$, the BDD eventually converges to a value corresponding to the set of all V-atoms reachable from V-atoms which satisfy $init$. The following set denotes this:

$$\{\beta \in Atoms_V : \text{ for some } \alpha \in Atoms_V, \ \dashv (\$T)^* \wedge \alpha \wedge init \wedge sfin\ \beta\} \ .$$

We then terminate the algorithm with a report that the original transition configuration $\Box T \wedge init \wedge finite$ is unsatisfiable. Even though Theorem 38 bounds the number of iterations, in some cases convergence takes too long. This necessitates a preset iteration limit or a facility for manual intervention in order to force premature termination of the loop.

If for some n, the algorithm succeeds after n iterations and determines that the transition configuration is satisfiable, then a sample V-interval having $n+1$ states and which satisfies the formula can be calculated. This involves standard BDD methods for constructing such examples and is done by working backward through the BDDs Δ_n, Δ_{n-1}, $\ldots \Delta_0$ to find a suitable sequence of $n+1$ V-atoms to serve as a V-interval satisfying the transition configuration. The algorithm can be also readily adapted to only determine values for a subset of the variables in V.

8.1 Dealing with Infinite Time

For testing an infinite-time transition configuration $\Box T \wedge init \wedge \Box \Diamond^+ L$, we can make use of Theorem 36 which guarantees that this formula is satisfiable iff the next PTL_V formula is satisfiable:

$$\boxdot T \wedge init \wedge \Diamond(L \wedge finite \wedge more \wedge (\vec{V} \leftarrow \vec{V})) \ .$$

The previously described satisfiability algorithm for finite-time can therefore be utilised. However, we must first transform this second formula to some suitable finite-time transition configuration using techniques later described in Sect. 12 for reducing arbitrary PTL formulas to finite-time transition configurations. Alternatively, more sophisticated algorithms using Theorem 49 can be employed to directly analyse the infinite-time transition configuration using BDD-based techniques. Space does not permit more details here.

9 Axiom System for NL

In preparation for the proof of axiomatic completeness for PTL, we now consider an axiom system for NL. The axiomatic completeness of NL later plays a major role in the completeness proof for PTL.

Within this section, the variables X, X', X_0 and X'_0 denote NL *formulas.*

Table 5 contains a complete axiom system for NL adapted from the modal logic K+D$_c$. Here ⓦ ("weak next"), previously defined in Table 1 to be a derived operator, is instead regarded as a primitive construct. We can consider $\bigcirc X$ to be an abbreviation for $\neg$ⓦ$\neg X$. Hughes and Cresswell [36, Problem 6.8 on p. 123 with solution on p. 379] briefly discuss how to show deductive completeness of the logic K+D$_c$.

Axioms:

N1 (K).	$\vdash \text{ⓦ}(X \supset X') \supset \text{ⓦ}X \supset \text{ⓦ}X'$
N2 ($\mathrm{D_c}$).	$\vdash \bigcirc X \supset \text{ⓦ}X$

Inference rules:

NR1.	If X is a tautology, then $\vdash X$
NR2 (MP).	If $\vdash X \supset X'$ and $\vdash X$, then $\vdash X'$
NR3 (RN).	If $\vdash X$, then $\vdash \text{ⓦ}X$

Table 5: Complete axiom system for NL (Modal system K+$\mathrm{D_c}$)

Axioms:

N1′ (N◇).	$\vdash \neg\bigcirc \mathit{false}$
N2′ (C◇).	$\vdash \bigcirc(X \vee X') \supset \bigcirc X \vee \bigcirc X'$
N3′ ($\mathrm{D_c}$).	$\vdash \bigcirc X \supset \text{ⓦ}X$

Inference rules:

NR1′.	If X is a tautology, then $\vdash X$
NR2′ (MP).	If $\vdash X \supset X'$ and $\vdash X$, then $\vdash X'$
NR3′ (RM◇).	If $\vdash X \supset X'$, then $\vdash \bigcirc X \supset \bigcirc X'$

Table 6: Alternative complete axiom system for NL based on $\bigcirc$

Table 6 contains a complete axiom system for NL in which $\bigcirc$, rather than ⓦ, is the primitive operator. Consequently, ⓦ is derived in the manner already shown in Table 1. The axiom system is essentially one of several M-based axiomatisations of normal systems of modal logic covered by Chellas [12] with the addition of the axiom $\mathrm{D_c}$. This second axiom system appears preferable for our purposes since our definition of PTL also takes $\bigcirc$ to be primitive. We therefore use this axiom system here although the methods employed can be easily adapted to the first NL axiom system.

Definition 53 (Theoremhood and Consistency for NL) *If some* NL *formula X is deducible from the axiom system, we call it an* NL theorem *and denote this theoremhood as $\vdash_{\mathrm{NL}} X$. We define X to be* NL-consistent *if $\neg X$ is* not *an* NL *theorem, i.e., $\nvdash_{\mathrm{NL}} \neg X$.*

Below are some representative lemmas about satisfiability and consistency of NL formulas. They are subsequently used in the completeness proof for the NL axiom system in Table 6.

Lemma 54 *For any state formula w and* NL *formula X, if w is satisfiable, then the* NL *conjunction $w \wedge \neg\bigcirc X$ is satisfied by some one-state interval.*

Lemma 55 *For any state formula w and* NL *formula X, if both w and X are satisfiable, then so is the formula $w \wedge \bigcirc X$.*
In such as case, if X itself is satisfied by an interval having at most n states, then $w \wedge \bigcirc X$ is satisfied by an interval having at most $n+1$ states,

Lemma 56 *For any* NL *formula X, if $\bigcirc X$ is* NL*-consistent, then so X.*

For any NL formulas X and X', the following are deducible as NL theorems and shortly used to simplify formulas:

$$\vdash_{\mathrm{NL}} \bigcirc(X \wedge X') \equiv \bigcirc X \wedge \bigcirc X' \tag{16}$$

$$\vdash_{\mathrm{NL}} \bigcirc(X \wedge \neg X') \equiv \bigcirc X \wedge \neg\bigcirc X' \tag{17}$$

$$\vdash_{\mathrm{NL}} \neg\bigcirc(X \vee X') \equiv \neg\bigcirc X \wedge \neg\bigcirc X' \ . \tag{18}$$

Axiomatic completeness is usually defined to mean that every valid formula is deducible as a theorem. However, we will make use of the following variant way of expressing completeness:

Axioms:		Inference rules:	
T1.	$\vdash \Box(X \supset Y) \supset \Box X \supset \Box Y$	R1.	If X is a tautology, then $\vdash X$
T2.	$\vdash \bigcirc X \supset Ⓦ X$	R2.	If $\vdash X \supset Y$ and $\vdash X$, then $\vdash Y$
T3.	$\vdash \bigcirc(X \supset Y) \supset \bigcirc X \supset \bigcirc Y$	R3.	If $\vdash X$, then $\vdash \Box X$
T4.	$\vdash \Box X \supset X \wedge Ⓦ \Box X$		
T5.	$\vdash \Box(X \supset Ⓦ X) \supset X \supset \Box X$		

Table 7: Modified version of Pnueli's complete PTL axiom system DX

Lemma 57 (Alternative Notion of Completeness) *A logic's axiom system is complete iff each consistent formula is satisfiable.*

Theorem 58 (Completeness of Alternative NL Axiom System) *The* NL *axiom system in Table 6 is complete.*

Proof The proof involves the kind of consistency-based reasoning found later in the paper. Using Lemma 57, we show that any NL formula X_0 which is NL-consistent (i.e., $\nvdash_{\mathrm{NL}} \neg X_0$) has a satisfying finite interval. Let n be the next-height of X_0, i.e., the maximum nesting of $\bigcirc$s in X_0. We do induction on n to show that X_0 is satisfied by some interval with at most $n+1$ states. □

10 Axiomatic Completeness for Transition Configurations

We now turn to describing a PTL axiom system with which axiomatic completeness can be shown for transition configurations.

The PTL axiom system used here is shown in Table 7 and is adapted from another similar PTL axiom system DX proposed by Pnueli [61]. Gabbay et al. [27] showed that DX is complete. Pnueli's original system uses strong versions of $\Diamond$ and $\Box$ (which we denote as $\Diamond^+$ and $\Box^+$, respectively) which do not examine the current state. In addition, Pnueli's system only deals with infinite time. However, Gabbay et al. [27] also include a variant system called $\mathrm{D^0X}$ based on the conventional $\Diamond$ and $\Box$ operators which examine the current state. The version presented here does this as well and furthermore permits both finite and infinite time.

Definition 59 (Theoremhood and Consistency for PTL) *If the* PTL *formula X is deducible from the axiom system, we call it a* PTL theorem *and denote this theoremhood as $\vdash X$. We define X to be* consistent *if $\neg X$ is* not *a theorem, i.e., $\nvdash \neg X$.*

In the course of proving completeness for PTL we make use of a definition of completeness for sets of formulas such as sets of transitions configurations:

Definition 60 (Completeness for a Set of Formulas) *An axiom system is said to be complete for a set of formulas $\{X_1, \ldots, X_n\}$ if the consistency of any X_i implies that X_i is also satisfiable.*

Now the Alternative Notion of Completeness (Lemma 57) can also be readily adapted to sets of formulas. Indeed, our goal in the rest of this section is to show that any consistent transition configuration is also satisfiable.

The next lemma permits us to utilise within PTL the axiomatic completeness of the NL proof system:

Theorem 61 (Completeness for NL in PTL) *The* PTL *axiom system is complete for the set of* NL *formulas.*

Proof Theorem 58 establishes the completeness of the alternative NL axiom system in Table 6. We then show that any NL theorem is also a PTL theorem. This can be done by demonstrating that all axioms and inferences rules in the NL axiom system are derivable from PTL ones. □

Lemma	Summary
62	If $\dashv\ \boxed{\text{m}}\, T \wedge \alpha \wedge \bigcirc \beta$, then $\Dashv\ T \wedge \alpha \wedge \mathit{skip} \wedge \mathit{sfin}\ \beta$
64	If $\dashv\ \boxed{\text{m}}\, T \wedge \alpha \wedge \Diamond \beta$, then $\Dashv\ (\$\, T)^* \wedge \alpha \wedge \mathit{sfin}\ \beta$
65	If $\dashv\ \boxed{\text{m}}\, T \wedge \alpha \wedge \Diamond^+ \beta$, then $\Dashv\ (\$\, T)^* \wedge \alpha \wedge \bigcirc\, \mathit{sfin}\ \beta$

Table 8: Summary of some basic lemmas for consistency and satisfiability

10.1 Some Basic Lemmas for Completeness

In this subsection, we deal with another part of the completeness proof. We utilise ways to go from certain specific kinds of consistent formulas involving reachability to intervals in order to later construct models for consistent transition configurations in §10.2. Table 8 summarises the basic lemmas proved here. Within the table, we use the notation $\Dashv X$ already introduced in Definition 1 to denote that the formula X is satisfiable and $\dashv X$ to denote that X is consistent.

Lemma 62 *For any V-atoms α and β, if the formula $\boxed{\text{m}}\, T \wedge \alpha \wedge \bigcirc \beta$ is consistent, then the formula $T \wedge \alpha \wedge \mathit{skip} \wedge \mathit{sfin}\ \beta$ is satisfiable.*

Proof From the consistency of the formula $\boxed{\text{m}}\, T \wedge \alpha \wedge \bigcirc \beta$ and simple temporal reasoning, we obtain the consistency of the NL^1_V formula $T \wedge \alpha \wedge \bigcirc \beta$. Theorem 61 concerning axiomatic completeness for NL formulas in the PTL axiom system then ensures that this is satisfiable. Clearly any interval satisfying it has at least two states. Hence by the earlier Lemma 50 the formula $\mathit{skip} \wedge T \wedge \alpha \wedge \bigcirc \beta$ is also satisfiable. Consequently, simple temporal reasoning yields that the semantically equivalent formula $T \wedge \alpha \wedge \mathit{skip} \wedge \mathit{sfin}\ \beta$ is satisfiable as well. □

For any V-atom α, within the next two lemmas we let S_α denote the subset of Atoms_V containing exactly every V-atom γ for which the following formula, which concerns reachability from α, is satisfiable:

$$(\$\, T)^* \wedge \alpha \wedge \mathit{sfin}\ \gamma \ .$$

Here is a more formal definition of S_α:

$$S_\alpha \quad \stackrel{\text{def}}{=} \quad \{\, \gamma \in \mathit{Atoms}_V :\ \Dashv (\$\, T)^* \wedge \alpha \wedge \mathit{sfin}\ \gamma \,\} \ .$$

Lemma 63 *For any V-atom α, the following formula is a* PTL *theorem:*

$$\vdash \quad \boxed{\text{m}}\, T \wedge \alpha \supset \Box \bigvee_{\gamma \in S_\alpha} \gamma \ . \tag{19}$$

Proof The following formulas are valid and in NL^1. Hence, they are theorems by the completeness of the PTL axiom system for NL^1 formulas (Theorem 61):

$$\vdash \quad \alpha \supset \bigvee_{\gamma \in S_\alpha} \gamma \qquad\qquad \vdash \quad \mathit{more} \wedge T \wedge \bigvee_{\gamma \in S_\alpha} \gamma \quad \supset \quad \bigcirc \bigvee_{\gamma \in S_\alpha} \gamma \ .$$

From these and simple temporal reasoning we can readily deduce our goal (19). □

Lemma 64 *For any V-atoms α and β, if the formula $\boxed{\text{m}}\, T \wedge \alpha \wedge \Diamond \beta$ is consistent, then the formula $(\$\, T)^* \wedge \alpha \wedge \mathit{sfin}\ \beta$ is satisfiable.*

Proof Suppose on the contrary that $(\$\, T)^* \wedge \alpha \wedge \mathit{sfin}\ \beta$ is unsatisfiable. Now α is in the set S_α, whereas β is not. Hence, the following formula concerning β not being in S_α is valid and thus a propositional tautology:

$$\vdash \quad \bigvee_{\gamma \in S_\alpha} \gamma \quad \supset \quad \neg \beta \ . \tag{20}$$

Furthermore, the previous Lemma 63 ensures that the next implication is a PTL theorem:

$$\vdash \quad \boxed{\scriptstyle m}\, T \wedge \alpha \supset \Box \bigvee_{\gamma \in S_\alpha} \gamma \ . \tag{21}$$

The two implications (20) and (21) together with some simple temporal reasoning let us deduce that α can never reach β:

$$\vdash \quad \boxed{\scriptstyle m}\, T \wedge \alpha \supset \Box \neg\beta \ .$$

From this and the general equivalence $\vdash \Box \neg\beta \equiv \neg \Diamond \beta$ we can deduce the following PTL theorem:

$$\vdash \quad \boxed{\scriptstyle m}\, T \wedge \alpha \supset \neg \Diamond \beta \ .$$

Therefore, the formula $\boxed{\scriptstyle m}\, T \wedge \alpha \wedge \Diamond \beta$ is inconsistent. This contradicts the lemma's assumption. $\Box$

Lemma 65 *For any V-atoms α and β, if the formula $\boxed{\scriptstyle m}\, T \wedge \alpha \wedge \Diamond^+ \beta$ is consistent, then the formula $(\$T)^* \wedge \alpha \wedge \bigcirc \mathit{sfin}\ \beta$ is satisfiable.*

Proof From the consistency of the formula $\boxed{\scriptstyle m}\, T \wedge \alpha \wedge \Diamond^+ \beta$, we readily deduce for some V-atom γ the consistency of the two PTL_V formulas below:

$$\boxed{\scriptstyle m}\, T \wedge \alpha \wedge \Diamond \gamma \qquad \boxed{\scriptstyle m}\, T \wedge \gamma \wedge \bigcirc \beta \ .$$

The consistency of the first formula $\boxed{\scriptstyle m}\, T \wedge \alpha \wedge \Diamond \gamma$ and Lemma 65 yield that the formula $(\$T)^* \wedge \alpha \wedge \mathit{sfin}\ \gamma$ is satisfiable. Lemma 62 and the second formula $\boxed{\scriptstyle m}\, T \wedge \gamma \wedge \bigcirc \beta$ then guarantee that the formula $T \wedge \gamma \wedge \mathit{skip} \wedge \mathit{sfin}\ \beta$ is satisfiable. Lemma 31 then yields that the next formula is satisfiable:

$$\big((\$T)^* \wedge \alpha \wedge \mathit{finite}\big); (T \wedge \gamma \wedge \mathit{skip} \wedge \mathit{sfin}\ \beta) \ .$$

From this and some further simple interval-based reasoning we can establish our goal, namely, that the formula $(\$T)^* \wedge \alpha \wedge \bigcirc \mathit{sfin}\ \beta$ is satisfiable. $\Box$

10.2 Completeness for Transition Configurations

We now apply the material presented in the previous §10.1 to ultimately establish completeness for finite- and infinite-time transition configurations. Here is a summary of the completeness theorems for them:

Type of transition	Where proved
Finite-time	Theorem 66
Infinite-time	Theorem 67

The remaining two kinds of transition configurations are subordinate to these. For the sake of brevity, we do not consider them here.

Theorem 66 *Completeness holds for any finite-time transition configuration $\Box T \wedge \mathit{init} \wedge \mathit{finite}$.*

Proof From the consistency of the finite-time transition configuration $\Box T \wedge \mathit{init} \wedge \mathit{finite}$ and simple temporal reasoning we can demonstrate that for some V-atoms α and β, the next formula is consistent:

$$\boxed{\scriptstyle m}\, T \wedge \alpha \wedge \mathit{init} \wedge \mathit{sfin}\,(T \wedge \beta) \ .$$

From this and further simple temporal reasoning it is readily follows that the following formulas are all consistent:

$$\alpha \wedge \mathit{init} \qquad \boxed{\scriptstyle m}\, T \wedge \alpha \wedge \Diamond \beta \qquad T \wedge \beta \wedge \mathit{empty} \ .$$

The first of these is itself satisfiable since any consistent formula in PROP is satisfiable. The second one and Lemma 64 yields that the PITL formula $(\$T)^* \wedge \alpha \wedge \mathit{sfin}\ \beta$ is satisfiable. The third formula $T \wedge \beta \wedge \mathit{empty}$ is in NL^1 and hence by Theorem 61 satisfiable. Hence the following formulas are all satisfiable:

$$\alpha \wedge \mathit{init} \qquad (\$T)^* \wedge \alpha \wedge \mathit{sfin}\ \beta \qquad T \wedge \beta \wedge \mathit{empty} \ .$$

This and Theorem 46 then yield the satisfiability of the finite-time transition configuration $\Box T \wedge \mathit{init} \wedge \mathit{finite}$. $\Box$

Theorem 67 *Completeness holds for any infinite-time transition configuration* $\Box T \wedge \mathit{init} \wedge \Box \Diamond^+ L$.

Proof From the consistency of the infinite-time transition configuration $\Box T \wedge \mathit{init} \wedge \Box \Diamond^+ L$ and simple temporal reasoning we can demonstrate that for some V-atoms α and β, the next formula is consistent:

$$\boxed{\mathrm{m}}\, T \wedge \alpha \wedge \mathit{init} \wedge \Box \Diamond^+(\beta \wedge L) \quad . \tag{22}$$

Lemma 40 ensures that the formulas $\beta \wedge L$ and $\beta \wedge \mathit{En}_{L,\beta}$ are semantically equivalent. The proof of this only requires simple propositional reasoning not involving the temporal operators in L. Hence the next equivalence is readily deducible as a PTL theorem using substitution into a propositional tautology (see Definition 3 and PTL inference rule R1 in Table 7):

$$\vdash \quad \beta \wedge L \;\equiv\; \beta \wedge \mathit{En}_{L,\beta} \quad . \tag{23}$$

From the consistency of formula (22) and the deducibility of formula (23), we can show the consistency of the next formula:

$$\boxed{\mathrm{m}}\, T \wedge \alpha \wedge \mathit{init} \wedge \Box \Diamond^+(\beta \wedge \mathit{En}_{L,\beta}) \quad .$$

This and simple temporal reasoning then together yield the consistency of the following formulas involving some additional V-atoms $\gamma_1, \ldots, \gamma_{|\mathit{En}_{L,\beta}|}$ (not necessarily distinct):

$$\begin{array}{llll} \alpha \wedge \mathit{init} & \boxed{\mathrm{m}}\, T \wedge \alpha \wedge \Diamond \beta & \boxed{\mathrm{m}}\, T \wedge \beta \wedge \Diamond^+ \beta & \\ \text{for each } \gamma_i\colon & \boxed{\mathrm{m}}\, T \wedge \beta \wedge \Diamond \gamma_i & \gamma_i \wedge \theta_{\mathit{En}_{L,\beta}[i]} & \boxed{\mathrm{m}}\, T \wedge \gamma_i \wedge \Diamond \beta \quad . \end{array}$$

The consistency of the propositional formulas $\alpha \wedge \mathit{init}$ and $\gamma_i \wedge \theta_{\mathit{En}_{L,\beta}[i]}$ for each V-atom γ_i ensures they are satisfiable. Lemma 64 is then applied to the remaining consistent formulas, except for $\boxed{\mathrm{m}}\, T \wedge \beta \wedge \Diamond^+ \beta$ which requires Lemma 65. The combined result is that the following formulas are all satisfiable:

$$\begin{array}{llll} \alpha \wedge \mathit{init} & (\$T)^* \wedge \alpha \wedge \mathit{sfin}\ \beta & (\$T)^* \wedge \beta \wedge \bigcirc\, \mathit{sfin}\ \beta & \\ \text{for each } \gamma_i\colon & (\$T)^* \wedge \beta \wedge \mathit{sfin}\ \gamma_i & \gamma_i \wedge \theta_{\mathit{En}_{L,\beta}[i]} & (\$T)^* \wedge \gamma_i \wedge \mathit{sfin}\ \beta \quad . \end{array}$$

Hence by Theorem 49, the original consistent infinite-time transition configuration is indeed satisfiable. $\Box$

11 Invariants and Related Formulas

We will shortly introduce the concepts of invariants and invariant configurations which together act as a natural middle level between transition configurations and full PTL and involve the use of auxiliary variables. These variables provide a way to reduce the nesting of temporal operators within other temporal operators and thereby simplify further analysis. Satisfiability, existence of small models, decidability and axiomatic completeness for invariant configurations can be readily related to the analysis of transition configurations. Furthermore, it is not hard to reduce arbitrary PTL formulas to invariant configurations by utilising such auxiliary variables.

The analysis of invariant configurations and arbitrary PTL formulas does not require any further interval-based reasoning or PITL.

Definition 68 (Invariant) *An* invariant *is any finite conjunction of zero or more equivalences in which each equivalence's left side is a distinct propositional variable and each equivalence's right side is one of the following:*

- *Some* PTL *formula of the form* $\Diamond w$, *for some state formula* w.
- *Some* NL^1 *formula.*

The variables occurring on the left sides of equivalences are called *dependent variables* and any other variables are called *independent variables*. The right sides are called *dependent formulas* and each equivalence is itself called a *dependency*. Hence for a given invariant I, it follows that $|I|$ denotes the number of dependencies in I. Also, for any $k : 1 \leq k \leq |I|$, $I[k]$ denote the k-th dependency in I. Each dependency containing $\Diamond$ is referred to as a $\Diamond$-*dependency*. Observe that a dependent variable can be referenced in any dependent formula including the one associated with it.

Below is a sample invariant referred to as I_1:

$$I_1 : \ (r_1 \equiv \Diamond(p \wedge \neg q)) \ \wedge \ (r_2 \equiv (r_1 \wedge \bigcirc r_2)) \ .$$

Here $|I_1|$ equals 2, the first dependency $I[1]$ is the equivalence $r_1 \equiv \Diamond(p \wedge \neg q)$ and the second dependency $I[2]$ is the equivalence $r_2 \equiv (r_1 \wedge \bigcirc r_2)$.

Note that an invariant is not necessarily satisfiable as in $r_1 \equiv \neg r_1$. Also note that dependencies of the two forms $r \equiv w$ and $r \equiv \bigcirc w$, for some propositional variable r and state formula w, are both subsumed by the second case in Definition 68. If desired, a more restrictive definition of invariants limited to dependencies of the form w, $\bigcirc w$ and $\Diamond w$ is possible.

We can view an invariant I as being any conjunction having the form $\bigwedge_{k:1 \leq k \leq |I|}(u_k \equiv \phi_k)$ so that u_k is the k-th dependent propositional variable and ϕ_k is the k-th dependent formula in I. Observe that for any $k : 1 \leq k \leq |I|$, the conjunct $I[k]$ has the form $u_k \equiv \phi_k$ and I itself can be expressed as $\bigwedge_{k:1 \leq k \leq |I|} I[k]$.

Starting with an invariant I, we analyse certain low-level formulas referred to here as *invariant configurations*.

Definition 69 (Invariant Configurations) *An* invariant configuration *is a formula of the form* $\Box I \wedge X$ *where the* PTL *formula* X *is in one of three categories shown below:*

Type of invariant configuration	*Syntax of* X
Basic	w
Finite-time	$w \wedge \mathit{finite}$
Infinite-time	$w \wedge \mathit{inf}$

Here w *is a state formula.*

For example, the conjunction $\Box I_1 \wedge r_2$ is a basic invariant configuration which is true for intervals which are infinite, have r_1 and r_2 always true and p and $\neg q$ both always eventually true.

The next definition helps to simplify the notation used in the reduction of invariant configurations to transition configurations:

Definition 70 (Ordered Invariant) *An invariant is said to be* ordered *if all of its* $\Diamond$*-dependencies precede any others.*

It is not hard to rearrange an arbitrary invariant's dependencies to obtain a semantically equivalent ordered invariant. In the rest of this section, we will without loss of generality limit our attention to ordered invariants and invariant configurations based on them.

We now associate with an ordered invariant I a transition formula T_I and a conditional liveness formula L_I. They serve to expeditiously reduce invariant configurations to transition configurations previously analysed in earlier sections. Definition 71 below describes T_I. The subsequent Definition 73 describes the form of L_I.

Definition 71 (Transition Formula for an Ordered Invariant) *For an ordered invariant* I*, the associated transition formula* T_I *is an* NL^1 *formula which captures* I*'s transitional behaviour between pairs of adjacent states. It is obtained from* I *by replacing each* $\Diamond$*-dependency with another dependency not containing* $\Diamond$ *and leaving the remaining* $\Diamond$*-free dependencies unchanged. More precisely, each dependency in* I *of the form* $r \equiv \Diamond w$*, for some propositional variable* r *and state formula* w*, is replaced by the* $\Diamond$*-free equivalence* $r \equiv (w \vee \bigcirc r)$*.*

Observe that the transition formula T_I is in NL^1 and is also a well-formed invariant. Also, for any $k : 1 \leq k \leq |I|$, if the dependency $I[k]$ does not contain $\Diamond$, then it and T_I's corresponding dependency $T_I[k]$ are identical.

Here is the transition formula T_{I_1} associated with I_1:

$$T_{I_1}: \ \big(r_1 \equiv ((p \wedge \neg q) \vee \bigcirc r_1)\big) \ \wedge \ (r_2 \equiv (r_1 \wedge \bigcirc r_2)) \ .$$

Let us now introduce some simple notation needed for reasoning about liveness and $\Diamond$-dependencies. This will be used in the definition of an ordered invariant's associated conditional liveness formula.

Definition 72 (Liveness Tests of an Ordered Invariant) *For any ordered invariant I having n $\Diamond$-dependencies, define n different* liveness tests $\theta_{I[1]}, \ \ldots, \ \theta_{I[n]}$ *so that for each $k : 1 \le k \le n$, the k-th dependency in I is expressible as $u_k \equiv \Diamond \theta_{I[k]}$.*

For instance, the sample invariant I_1 has a single liveness test $\theta_{I[1]}$ which denotes the formula $p \wedge \neg q$. Note that each $\theta_{I[k]}$ is always a state formula. If an invariant I has n $\Diamond$-dependencies, then for each $k : 1 \le k \le n$, T_I's dependency $T_I[k]$ identical to the equivalence $u_k \equiv (\theta_{I[k]} \vee \bigcirc u_k)$.

Given an ordered invariant I, we now associate a specific conditional liveness formula L_I with it:

Definition 73 (Conditional Liveness Formula of an Ordered Invariant) *The conditional liveness formula L_I of an ordered invariant I which has n $\Diamond$-dependencies is itself a conjunction of n implications. For each $k : 1 \le k \le n$, the k-th implication is obtained by simply replacing the outermost equivalence operator in I's k-th $\Diamond$-dependency by the implication operator and using $\Diamond\!\!\!\!\!\!{\scriptscriptstyle \mathrm{m}}\;\,$ instead of $\Diamond$. Therefore, for each $k : 1 \le k \le n$, the dependency $I[k]$ has the form $u_k \equiv \Diamond \theta_{I[k]}$ and the implication $L_I[k]$ has the form $u_k \supset \Diamond\!\!\!\!\!\!{\scriptscriptstyle \mathrm{m}}\;\, \theta_{I[k]}$.*

The definition of I's conditional liveness formula L_I intentionally ignores any NL^1 dependencies in I since T_I already adequately deals with them. As a result, L_I can contain fewer conjuncts than I and T_I. Below is the conditional liveness formula L_{I_1} associated with ordered invariant I_1:

$$L_{I_1}: \ \ (r_1 \supset \Diamond\!\!\!\!\!\!{\scriptscriptstyle \mathrm{m}}\;\,(p \wedge \neg q)) \ .$$

It is not hard to see that, unlike I's transition formula, the conditional liveness formula associated with I is not a well-formed invariant.

11.1 Reduction of Basic Invariant Configurations

Starting with an ordered invariant I, let us now consider the relationship between its basic invariant configuration and the associated finite-time and infinite-time invariant configurations. This permits us to focus the remaining analysis on the two later kinds of invariant configurations.

Lemma 74 *A basic invariant configuration $\Box I \wedge w$ is satisfiable iff at least one of its associated finite-time and infinite-time invariant configurations is satisfiable.*

Proof This follows from the validity of the formula *finite* $\vee$ *inf* and simple propositional reasoning. $\Box$

The finite-time and infinite-time invariant configurations for the ordered invariant I each have a corresponding semantically equivalent transition configuration of the same kind as is now shown:

	Invariant configuration	Transition configuration	Where proved
Finite time	$\Box I \wedge w \wedge \mathit{finite}$	$\Box T_I \wedge w \wedge \mathit{finite}$	Theorem 76
Infinite time	$\Box I \wedge w \wedge \mathit{inf}$	$\Box T_I \wedge w \wedge \Box \Diamond^{+} L_I$	Theorem 79

Observe that the reductions from the two types of the invariant configurations to the corresponding transition configurations do not introduce any extra variables. In what follows we prove that a finite-time invariant configuration is semantically equivalent to its associated finite-time transition configuration and similarly a infinite-time invariant configuration is semantically equivalent to its associated infinite-time transition configuration.

In what follows we will often abstract the behaviour of a $\Diamond$-dependency by using two propositional variables p and q and representing the dependency as the PTL equivalence $p \equiv \Diamond q$. This technique is used to establish the next lemma:

Lemma 75 *The formulas $\Box I$ and $\Box T_I$ are semantically equivalent on finite intervals. In other words, the following implication is valid:*

$$\models \quad \mathit{finite} \quad \supset \quad \Box I \;\equiv\; \Box T_I \;.$$

PROOF We can represent $\Box I$ as the conjunction $\bigwedge_{k:1\le k\le |I|} \Box I[k]$ and similarly represent $\Box T_I$ as the conjunction $\bigwedge_{k:1\le k\le |I|} \Box T_I[k]$. For any $k : 1 \le k \le |I|$, if $I[k]$ is in NL^1 then $T_I[k]$ is identical to it and hence $\Box I[k]$ and $\Box T_I[k]$ are identical. Otherwise, $\Box I[k]$ can be seen as a substitution instance of the PTL formula $\Box(p \equiv \Diamond q)$ containing the two propositional variables p and q. Now $\Box T_I[k]$ therefore corresponds to the formula $\Box(p \equiv (q \vee \bigcirc p))$. Simple temporal reasoning can then be used to show that each of these implies the other in any finite interval. $\Box$

Let us note that the validity for finite time of the relevant equivalence $\Box(p \equiv \Diamond q) \equiv \Box(p \equiv (q \vee \bigcirc p))$ can even be readily checked by a computer implementation of a decision procedure for PTL with finite time.

Theorem 76 *The finite-time invariant configuration for I is semantically equivalent to the associated finite-time transition configuration.*

PROOF This readily follows from Lemma 75 and propositional reasoning. $\Box$

Unfortunately, the equivalence $\Box I \equiv \Box T_I$ can fail to be valid for infinite time if I contains $\Diamond$-dependencies because T_I does not fully capture the liveness requirements of such dependencies. Lemma 78 later on corrects for this problem by showing that in infinite time the two formulas $\Box I$ and $\Box T_I \wedge \Box \Diamond^+ L_I$ are semantically equivalent. The reason that $\Box I \equiv \Box T_I$ is not necessarily valid is because when we consider an individual $\Diamond$-dependency, the formulas $\Box(p \equiv \Diamond q)$ and $\Box(p \equiv (q \vee \bigcirc p))$ are not semantically equivalent on infinite-time intervals since on such an interval, the first formula can be false and the second one true. An example of this occurs in any infinite interval where p is always true and q is always false. Therefore, if I contains $\Diamond$-dependencies, then $\Box I$ can be false on an infinite-time interval even though $\Box T_I$ is true on the interval. However, the next lemma holds even for infinite time:

Lemma 77 *The* PTL *implication $\Box I \supset \Box T_I$ is valid.*

PROOF The NL^1-dependencies in I and T_I are identical. Furthermore, for the $\Diamond$-dependencies we make use of the valid PTL formula $\Box(p \equiv \Diamond q) \supset \Box(p \equiv (q \vee \bigcirc p))$. $\Box$

We see from Lemma 77 that the formula $\Box I \supset \Box T_I$ is valid for both finite and infinite time. However if I contains $\Diamond$-dependencies, then the converse implication $\Box T_I \supset \Box I$ is not necessarily valid for infinite time because the implication $\Box(p \equiv (q \vee \bigcirc p)) \supset \Box(p \equiv \Diamond q)$ fails to be valid. We now discuss the principles which successfully correct for this. First of all, the following weakened implication concerning an individual $\Diamond$-dependency is valid:

$$\models \quad \Box(p \equiv (q \vee \bigcirc p)) \quad \supset \quad \Box(\Diamond q \supset p) \;.$$

Here we use the formula $\Diamond q \supset p$ instead of the stronger equivalence $p \equiv \Diamond q$. The following equivalence then strengthens the effect of $\Box(p \equiv (q \vee \bigcirc p))$ by adding the formula $\Box(p \supset \Diamond q)$:

$$\models \quad \Box(p \equiv \Diamond q) \quad \equiv \quad \Box(p \equiv (q \vee \bigcirc p)) \wedge \Box(p \supset \Diamond q) \;.$$

In fact, we can even replace the conjunct $\Box(p \supset \Diamond q)$ by the weaker formula $\Box\Diamond(p \supset \Diamond q)$ which adds a $\Diamond$:

$$\models \quad \Box(p \equiv \Diamond q) \quad \equiv \quad \Box(p \equiv (q \vee \bigcirc p)) \wedge \Box\Diamond(p \supset \Diamond q) \;.$$

All three valid formulas only contain the propositional variables p and q and can consequently be readily checked for infinite-time validity by any computer implementation of a decision procedure for PTL with infinite time.

Now suppose the ordered invariant I has m $\Diamond$-dependencies and hence $m = |L_I|$. If we have m pairs of propositional variables $p_1,\ q_1,\ \ldots,\ p_m,\ q_m$ (corresponding to I's $\Diamond$-dependencies) then the following generalisation of the previous valid equivalence is itself valid:

$$\models \quad \Box \bigwedge_{1\le k\le m} (p_k \equiv \Diamond q_k)$$

$$\equiv \quad \Box \bigwedge_{1\le k\le m} (p_k \equiv (q_k \vee \bigcirc p_k)) \ \wedge \ \Box\Diamond \bigwedge_{1\le k\le m} (p_k \ \supset \ \Diamond q_k) \ .$$

The left side of the equivalence corresponds to the invariant $I[1:m]$. Similarly, the first conjunct on the right side corresponds to $T_I[1:m]$ and the second one to L_I, except for the use of $\Diamond$ instead of $\Diamond^{+}$.

Now within infinite time, $\Box\Diamond$ and $\Box\Diamond^{+}$ have the same behaviour and in addition $\Diamond$ and $\Diamond^{+}$ act identically. We use this to obtain the next lemma which expresses I in terms of T_I and L_I:

Lemma 78 *The formula* $inf \ \supset \ \big(\Box I \equiv (\Box T_I \wedge \Box\Diamond^{+} L_I)\big)$ *is valid.*

Theorem 79 *An infinite-time invariant configuration* $\Box I \wedge w \wedge inf$ *for the ordered invariant* I *is semantically equivalent to the associated infinite-time transition configuration* $\Box T_I \wedge w \wedge \Box\Diamond^{+} L_I$*.*

Proof This readily follows from Lemma 78 and simple temporal reasoning. □

The soundness of the reductions to the associated transition configurations ensures that we can use the decision procedure described in Sect. 8.

11.2 Bounded Models for Basic Invariant Configurations

The theorem given below gives the small model property for basic invariant configurations:

Theorem 80 *Suppose* V *is a finite set of variables and the variables in the ordered invariant* I *and the state formula* w *are all elements of* V*. Then the basic invariant configuration* $\Box I \wedge w$ *is satisfiable iff it is satisfied by some some finite interval with interval length less than* $|Atoms_V|$ *or by an infinite, ultimately periodic one consisting of an initial segment with interval length at most* $|Atoms_V|$ *fused with a remaining infinite periodic part with a period having interval length at most* $(|L_I| + 1)\,|Atoms_V|$*.*

Proof Suppose $\Box I \wedge w$ is satisfiable. We will consider the two cases of finite and infinite intervals separately:

Case for finite intervals: Theorem 76 ensures that the finite-time invariant configuration $\Box I \wedge w \wedge finite$ and its associated finite-time transition configuration $\Box T_I \wedge w \wedge finite$ are semantically equivalent. The construction of T_I ensures that any variable occurring in it is a member of the set V. Lemma 38 therefore establishes that if the conjunction $\Box T_I \wedge w \wedge finite$ is satisfiable, then a satisfying interval exists having less interval length than $|Atoms_V|$. This interval consequently also satisfies the basic invariant configuration $\Box I \wedge w$.

Case for infinite intervals: Theorem 79 ensures that the infinite-time invariant configuration $\Box I \wedge w \wedge inf$ and its associated infinite-time transition configuration $\Box T_I \wedge w \wedge \Box\Diamond^{+} L_I$ are semantically equivalent. From Lemma 45 we have that this second formula is satisfied by an infinite interval consisting of an initial segment having interval length less than $|Atoms_V|$ fused with a periodic interval with period having interval length at most $(|L_I|+1)\,|Atoms_V|$. The overall ultimately periodic interval therefore also satisfies the formula $\Box I \wedge w$. □

11.3 Axiomatic Completeness for Invariant Configurations

Theorem 81 *Completeness holds for finite- and infinite-time invariant configurations.*

Proof Suppose we have some invariant I. Assume without loss of generality that I is ordered since otherwise we can trivially rearrange its dependencies to obtain an ordered invariant which is both semantically and deducibly equivalent to I. Subsection 11.1 already described how to construct a semantically equivalent transition configuration from any finite-time or infinite-time invariant configuration associated with I.

The various valid formulas mentioned there can be deduced as PTL theorems to establish that each such finite-time and infinite-time invariant configuration is also deducibly equivalent to the associated transition configuration. This and the previously shown axiomatic completeness for finite-time and infinite-time transition configurations respectively proved in Theorems 66 and 67 ensure that any consistent finite-time or infinite-time invariant configuration associated with I is satisfiable. Hence, we establish our immediate goal of completeness for finite- and infinite-time invariant configurations. $\square$

Theorem 82 *Completeness holds for basic invariant configurations.*

Proof Suppose we have some consistent basic invariant configuration $\Box I \wedge w$. Now the disjunction *finite* $\vee$ *inf* is easily deduced as a propositional tautology since *inf* is defined to be $\neg$*finite* (see Table 1). It is then straightforward to show using purely propositional reasoning that $\Box I \wedge w$ is deducibly equivalent to the disjunction of its associated finite-time or infinite-time invariant configurations:

$$\vdash \quad \Box I \wedge w \quad \equiv \quad (\Box I \wedge w \wedge \mathit{finite}) \vee (\Box I \wedge w \wedge \mathit{inf}) \ .$$

Hence at least one of the latter is also consistent. The previous Theorem 81 ensures that any such consistent finite- or infinite-time invariant configuration is satisfiable as well. An interval which satisfies it can also serve as a model for the basic invariant configuration. This demonstrates the desired axiomatic completeness for all basic invariant configurations. $\square$

12 Dealing with Arbitrary PTL Formulas

So far we have only looked at bounded models and axiomatic completeness for certain kinds of PTL formulas. For an arbitrary PTL formula X, it is straightforward to construct an invariant I linearly bounded by the size of X and containing a finite number of dependent variables $u_1, u_2, \ldots, u_{|I|}$ not themselves occurring in X so as to mimic the semantics of X in the sense that X is satisfiable iff $\Box I \wedge u_{|I|}$ is satisfiable and in addition the implication $\Box I \supset (u_{|I|} \equiv X)$ is valid.

One possible translation will be detailed shortly. Before describing it, we need to discuss a convention for systematically renaming an invariant's dependent variables. Normally, the first dependent variable in an invariant I constructed here from an PTL formula is r_1 and the last is $r_{|I|}$. However, we inductively construct the invariants by combining smaller invariants into larger ones and often must alter the indices of the dependent variables to avoid clashes. A operator on formulas to suitably do this is now defined:

Definition 83 (Shifting of Subscripts in Invariants) *For any invariant I, the operation $I\!\uparrow\! k$ is defined to be the invariant obtained by replacing $u_1, \ldots, u_{|I|}$ by $r_{1+k}, \ldots, r_{|I|+k}$, i.e., $I^{r_{1+k},\ldots,r_{|I|+k}}_{u_1,\ldots,u_{|I|}}$.*

It is not hard to see that if I's dependent variables are themselves the distinct variables $r_1, \ldots, r_{|I|}$, then $I\!\uparrow\! k$ shifts the subscripts of them so that each r_j becomes r_{j+k}. Therefore, the first dependent variable becomes r_{1+k} instead of r_1, the second becomes r_{2+k} and so forth. In other words, $I\!\uparrow\! k$ denotes the same formula as the conjunction $\bigwedge_{1\le j\le |I|}(r_{j+k} \equiv (\phi_j)^{r_{1+k},\ldots,r_{|I|+k}}_{u_1,\ldots,u_{|I|}})$.

Without loss of generality, let X be a PTL formula which does not contain any of the variables $r_1, r_2, \ldots$. Table 9 contains the definition of a function $\mathcal{H}(X)$ which translates X into an invariant containing some of the variables $r_1, r_2, \ldots$ as dependent variables. In order to reduce the number of dependent variables, the first case is used whenever the formula is in NL^1 even if one of the next two cases for negation and logical-or is applicable.

Table 10 contains a sample PTL formula X_0, an equivalent formula X_0' having no logical-ands, implications or $\Box$ constructs, and the invariant $\mathcal{H}(X_0')$ and the initial condition $r_{|\mathcal{H}(X_0')|}$. We also include a version of X_0' which shows how the dependencies correspond to the subformulas in X_0'.

It is straightforward to utilise more sophisticated methods which construct invariants directly from formulas with other logical operators such as logical-and and $\Box$. In addition, it is not hard to systematically produce invariants containing a lot fewer dependencies then the ones generated by $\mathcal{H}$. In fact, our prototype implementation of the decision procedure described in Sect. 8 makes use of such techniques and others as

X	$\mathcal{H}(X)$
T	$r_1 \equiv T$, for any NL^1 formula T.
$\neg Y$	$\mathcal{H}(Y) \ \wedge\ (r_{\lvert\mathcal{H}(Y)\rvert+1} \equiv \neg r_{\lvert\mathcal{H}(Y)\rvert})$
$Y \vee Y'$	$\mathcal{H}(Y) \ \wedge\ \mathcal{H}(Y') \uparrow m \ \wedge\ (r_{m+n+1} \equiv r_m \vee r_{m+n})$, where $m = \lvert\mathcal{H}(Y)\rvert$ and $n = \lvert\mathcal{H}(Y')\rvert$.
$\Diamond Y$	$\mathcal{H}(Y) \ \wedge\ (r_{\lvert\mathcal{H}(Y)\rvert+1} \equiv \Diamond r_{\lvert\mathcal{H}(Y)\rvert})$

Table 9: Definition of $\mathcal{H}(X)$

X_0: $\Box\big((p \supset \bigcirc\Diamond q) \ \wedge\ \Diamond(\neg p \vee \bigcirc q)\big)$

X_0': $\neg\Diamond\neg\Big(\neg(\neg(\neg p \vee \bigcirc\Diamond q) \ \vee\ \neg\Diamond(\neg p \vee \bigcirc q))\Big)$

X_0' with dependent variables shown:

$$\overbrace{\neg\underbrace{\Diamond\overbrace{\neg\underbrace{\Big(\neg\overbrace{(\underbrace{\neg(\overbrace{\underbrace{\neg p}_{r_1} \vee \underbrace{\bigcirc\overbrace{\Diamond\underbrace{q}_{r_2}}^{r_3}}_{r_4}}^{r_5})}_{r_6} \vee \underbrace{\neg\overbrace{\Diamond\underbrace{(\neg p \vee \bigcirc q)}_{r_7}}^{r_8}}_{r_9})}^{r_{10}}\Big)}_{r_{11}}}^{r_{12}}}_{r_{13}}}^{r_{14}}$$

$\mathcal{H}(X_0')$:

$$\begin{aligned}&(r_1 \equiv \neg p) \ \wedge\ (r_2 \equiv q) \ \wedge\ (r_3 \equiv \Diamond r_2) \ \wedge\ (r_4 \equiv \bigcirc r_3)\\ &\wedge\ (r_5 \equiv (r_1 \vee r_4)) \ \wedge\ (r_6 \equiv \neg r_5) \ \wedge\ (r_7 \equiv (\neg p \vee \bigcirc q))\\ &\wedge\ (r_8 \equiv \Diamond r_7) \ \wedge\ (r_9 \equiv \neg r_8) \ \wedge\ (r_{10} \equiv (r_6 \vee r_9))\\ &\wedge\ (r_{11} \equiv \neg r_{10}) \ \wedge\ (r_{12} \equiv \neg r_{11}) \ \wedge\ (r_{13} \equiv \Diamond r_{12})\\ &\wedge\ (r_{14} \equiv \neg r_{13})\end{aligned}$$

$r_{\lvert\mathcal{H}(X_0')\rvert}$: r_{14}

Table 10: Example of invariant obtained by applying $\mathcal{H}$ to a PTL formula

well. Here is an invariant and initial formula produced by the decision procedure directly from the formula X_0:

$$
\begin{array}{ll}
X_0: & \Box\big((p \supset \bigcirc \Diamond q) \wedge \Diamond(\neg p \vee \bigcirc q)\big) \\
I': & (r_1 \equiv \Diamond q) \wedge (r_2 \equiv \bigcirc r_1) \wedge (r_3 \equiv \bigcirc q) \wedge (r_4 \equiv \Diamond(\neg p \vee r_3)) \\
 & \quad \wedge \big(r_5 \equiv \Diamond \neg((p \supset r_2) \wedge r_4)\big) \\
\mathit{init}': & \neg r_5
\end{array}
$$

We omit further details.

It is easy to check that $\mathcal{H}(X)$ contains at most one dependent variable for each variable and operator in X so the total number of dependent variables in $\mathcal{H}(X)$ is bounded by X's size and indeed the size of $\mathcal{H}(X)$ is linearly bounded by X's size. It is also easy to check by doing induction on X's syntactic structure that X is satisfiable iff the basic invariant configuration $\Box\,\mathcal{H}(X) \wedge r_{|\mathcal{H}(X)|}$ is satisfiable. Furthermore, the implication $\Box\,\mathcal{H}(X) \supset (r_{|\mathcal{H}(X)|} \equiv X)$ can be shown to be valid. Consequently, $\Box\,\mathcal{H}(X) \wedge r_{|\mathcal{H}(X)|}$ is used to represent X's behaviour (modulo the dependent variables which act as auxiliary ones). The bounded model for the invariant configuration (see Theorem 80) satisfies X as well. The decision procedure described in Sect. 8 can be utilised to check the satisfiability of arbitrary PTL formulas by reducing them first to basic invariant configurations and then testing the associated finite-time and infinite-time transition configurations (see §11.1). Axiomatic completeness for X readily reduces to that for the invariant configuration $\Box\,\mathcal{H}(X) \wedge r_{|\mathcal{H}(X)|}$.

13 Some Additional Features

This section describes a number of extensions to our approach. They include the temporal operator *until* and past-time constructs and also a subset of PITL called *Fusion Logic* (FL) which includes constructs of the sort found in Propositional Dynamic Logic (PDL). In addition, the liveness tests found in conditional liveness formulas and invariants can be generalised to be of the form $\circledast T$, where T is an NL^1 formula, rather than just a state formula. We will consider each of these issues in turn. For the sake of brevity, the presentation is briefer and less formal than in the previous sections.

13.1 The Operator *until*

The operator *until* is a binary operator with the syntax $X\,\mathcal{U}\,Y$, where X and Y are PTL formulas. Recall from Sect. 4 that for any interval σ and natural number k which does not exceed σ's interval length, $\sigma_{k:|\sigma|}$ denotes the suffix subinterval obtained by deleting the first k states from σ. Here is the semantics of *until*:

$$\sigma \models X\,\mathcal{U}\,Y \quad \text{iff} \quad \text{for some } k \le |\sigma|,\ \sigma_{k:|\sigma|} \models Y \text{ and for all } j : 0 \le j < k, \sigma_{j:|\sigma|} \models X\ .$$

Observe that the operator $\Diamond$ can be expressed in terms of *until* since $\Diamond X$ is semantically equivalent to the formula *true until* X.

We can alter the definition of invariants by replacing $\Diamond$-dependencies with dependencies of the form $r \equiv (w\,\mathcal{U}\,w')$, where w and w' are state formulas. If the j-th dependency $I[j]$ of an invariant I is such a dependency (called an *until -dependency*), then the corresponding conjunction $T_I[j]$ in I's transition formula T_I has the form $r \equiv (w' \vee (w \wedge \bigcirc r))$. The associated conjunction $L_I[j]$ in L_I is $r \supset \circledast w'$. It is not hard to modify the material in Sect. 11 to ensure that finite-time and infinite-time invariant configurations remain semantically equivalent to the associated transition configurations.

Alternatively, we can transform an invariant with *until* in it to one without it. Each dependency in I of the form $u_k \equiv (w \textit{ until } w')$ is replaced by the dependency $u_k \equiv \big(u'_k \wedge (w' \vee (w \wedge \bigcirc u_k))\big)$, where u'_k is a new dependent variable with the associated dependency $u'_k \equiv \Diamond w'$. This approach is more hierarchical than the first one but increases the number of dependencies used.

13.2 Past Time

Let us now consider PTL with a bounded past. The syntax is modified to include the two additional primitive operators $\ominus X$ (read *previous* X) and $\diamondminus X$ (read *once* X). The set of PTL formulas including past-time constructs is denoted as $\mathrm{PTL}^{\mathrm{P}}$. The semantics of a PTL formula X is now expressed as $(\sigma, k) \models X$ where k is any natural number not exceeding $|\sigma|$. For example, the semantics of $\ominus$ and $\diamondminus$ are as follows:

$$
\begin{array}{lll}
(\sigma, k) \models \ominus X & \text{iff} & k > 0 \text{ and } (\sigma, k-1) \models X \\
(\sigma, k) \models \diamondminus X & \text{iff} & \text{for some } j : 0 \leq j \leq k, \ (\sigma, j) \models X \ .
\end{array}
$$

We define the operator $\boxminus X$ (read *so-far* X) as $\neg \diamondminus \neg X$ and the operator $\circleddash X$ (read *weak previous* X) as $\neg \ominus \neg X$. The operator *first* is defined to be $\neg \ominus\, \mathit{true}$ and tests for the first state of an interval. A past-time version of *until* called *since* can also be included but we omit the details.

A $\mathrm{PTL}^{\mathrm{P}}$ formula X is defined to satisfiable iff $(\sigma, k) \models X$ holds for some pair (σ, k) with $k \leq |\sigma|$. The formula X is valid iff $(\sigma, k) \models X$ holds for every pair (σ, k) with $k \leq |\sigma|$. Note that these straightforward definitions of satisfiability and validity correspond to the so-called *floating framework* of PTL with past time. However, Manna and Pnueli propose another interesting approach called the *anchored framework* [45] (also discussed in [43]) which they argue is superior. In this framework, satisfiability and validity only examine pairs of the form $(\sigma, 0)$. There exist ways to go between the two conventions but we will not delve into this here and instead simply assume the more traditional floating interpretation.

We now define an analogue of the set of formulas NL:

Definition 84 (Previous Logic) *The set of* PTL *formulas in which the only primitive temporal operator is* $\ominus$ *is called* Previous Logic *(*PrevL*). The subset of* PrevL *with no* $\ominus$ *nested in another* $\ominus$ *is denoted as* PrevL^1.

We let the variables Z and Z' denote formulas in PrevL^1. Also, PrevL^1_V denotes the set of all formulas in PrevL^1 only having variables in V.

The following definitions extend the notation of transition configurations to deal with past time:

Definition 85 (Past-Time Transition Configurations) *A* past-time transition configuration *is any formula of the form* $\boxminus\Box(T \wedge Z) \wedge X$*, where* T *is in* NL^1_V*,* Z *is in* PrevL^1_V*, and the formula* X *is in* PTL_V *and is in one of the two categories shown below:*

Type of configuration	*Syntax of X*
Finite-time	$w \wedge \mathit{finite}$
Infinite-time	$w \wedge \Box \Diamond^{+} L$

Here w *is a state formula in* PROP_V *and* L *is a conditional liveness formula in* PTL_V*.*

The formula $\boxminus\Box(T \wedge Z)$ contains both $\boxminus$ and $\Box$ to ensure that both T and Z are true everywhere in the interval.

The analysis of a finite-time or infinite-time past-time transition configurations can be easily reduced to reasoning in PTL without past time. Let us demonstrate this by first examining how to test the satisfiability of a finite-time past-time transition configuration $\boxminus\Box(T \wedge Z) \wedge w \wedge \mathit{finite}$. This involves finding an interval σ and natural number $k \leq |\sigma|$, such that $(\sigma, k) \models \boxminus\Box(T \wedge Z) \wedge w \wedge \mathit{finite}$ holds. Note that this past-time transition configuration is satisfiable iff the following formula, which shifts reasoning back to an interval's starting state, is satisfiable:

$$\diamondminus\big(\Box(T \wedge Z) \wedge \mathit{first} \wedge \Diamond w \wedge \mathit{finite}\big) \ . \tag{24}$$

Here we can dispense with the operator $\boxminus$ since $\boxminus\Box$ and $\Box$ have the same semantics at the starting state.

Now for any $\mathrm{PTL}^{\mathrm{P}}$ formula X, the formula $\diamondminus X$ is satisfiable iff X is satisfiable. Hence, the formula (24) is satisfiable iff its subformula $\Box(T \wedge Z) \wedge \mathit{first} \wedge \Diamond w \wedge \mathit{finite}$ is satisfiable. Let us now define the NL^1_V formula T' by replacing each $\ominus$ construct in Z by its operand and by taking each state formula in Z which does not occur in $\ominus$ and enclosing it in $\bigcirc$. For example, if Z is the formula $p \vee \ominus(q \wedge r)$, then T' is $(\bigcirc p) \vee (q \wedge r)$. Furthermore, let w' be the state formula in PROP_V obtained from Z by replacing each $\ominus$ construct by *false*. In our example, w' is $p \vee \mathit{false}$. It can be readily checked that the following formula relating Z and T' is true at any interval's initial state: $\Box Z \equiv \boxed{\scriptstyle\mathrm{m}}\, T' \wedge w'$. Therefore, the original finite-time past-time transition configuration is satisfiable iff the following formula in PTL without past time is satisfiable:

$$\Box(T \wedge (\mathit{more} \supset T')) \wedge w' \wedge \Diamond w \wedge \mathit{finite} \ . \tag{25}$$

This is still not a well-formed finite-time transition configuration due to the presence of the formula $\diamondminus\, w$. However, $\diamondminus\, w$ can be reduced by introducing a new propositional variable r as shown in the next formula:

$$\Box\big(T \wedge (\mathit{more} \supset T') \wedge (r \equiv (w \vee \bigcirc r))\big) \wedge w' \wedge r \wedge \mathit{finite} \ . \tag{26}$$

The reduction of the original past-time transition configuration $\boxminus\Box(T \wedge Z) \wedge w \wedge \mathit{finite}$ to the finite-time transition configuration (26) systematically relates all aspects of the analysis of the past-time transition configuration to the purely future-only reasoning presented earlier. This includes bounded models, decision procedures and axiomatic completeness.

An alternative way to reduce the PTL formula (25) involves interval-based reasoning. We first re-express the formula in PTL as the next semantically equivalent conjunction:

$$\boxed{\scriptstyle\mathrm{m}}(T \wedge (\mathit{more} \supset T')) \wedge w' \wedge \diamondminus\, w \wedge \mathit{sfin}\ T \ . \tag{27}$$

This makes use of the valid PTL equivalence $(\Box X \wedge \mathit{finite}) \equiv (\boxed{\scriptstyle\mathrm{m}}\, X \wedge \mathit{sfin}\ X)$, for any PTL formula X. However, in our case we can omit the subformula $\mathit{more} \supset T'$ in the sfin construct since the operator more ensures that it is trivially true in the associated empty interval. Let T'' denote the subformula $T \wedge (\mathit{more} \supset T')$. Theorem 11 ensures the semantic equivalence of $\boxed{\scriptstyle\mathrm{m}}\, T''$ and $(\$T'')^*$. Now the formula (27) can in turn be itself re-expressed as the following chop-formula:

$$((T'')^* \wedge w' \wedge \mathit{finite});\ ((T'')^* \wedge w \wedge \mathit{sfin}\ T) \ . \tag{28}$$

Let w'' denote a state formula obtained by replacing every $\bigcirc$ construct in T by false. Consequently, w'' is true exactly in states for which $T \wedge \mathit{empty}$ is true. It follows that we can test for satisfiability of formula (28) by adapting the symbolic methods mentioned in Sect. 8 to solve for V-atoms α, β and γ for which the following formulas are satisfiable:

$$\alpha \wedge w \qquad (\$T'')^* \wedge \alpha \wedge \mathit{sfin}\ \beta \qquad \beta \wedge w' \qquad (\$T'')^* \wedge \beta \wedge \mathit{sfin}\ \gamma \qquad \gamma \wedge w'' \ .$$

Further details are omitted here.

The treatment for a infinite-time past-time transition configuration is nearly identical to that for a finite-time one since the assumption of a bounded past still applies and avoids the need for a past-time conditional liveness formula. First of all, we replace the subformula finite by $\Box\Diamond^+ L$.

$$\Box(T \wedge T') \wedge w' \wedge \diamondminus\, w \wedge \Box\Diamond^+ L \ .$$

The use of infinite time ensures we can omit the instance of more found in the finite-time formulas (25) and (26) since T and $\mathit{more} \supset T$ are semantically equivalent on an infinite interval. The formula $\diamondminus\, w$ is itself reduced by introducing a new propositional variable r and conjoining a new implication to L to obtain the well-formed infinite-time transition configuration below:

$$\Box(T \wedge T') \wedge w' \wedge r \wedge \Box\Diamond^+(L \wedge (r \supset \Diamond_{\mathrm{m}}\, w)) \ .$$

So far we have only considered finite- and infinite-time transition configurations. Invariants (and hence also invariant configurations) can be extended to support past-time reasoning by adding two new kinds of dependencies. The first has the form $u \equiv Z$ and the second has the form $u \equiv \diamondminus\, w$. The use of $\diamondminus$ does not involve I's conditional liveness formula L_I due to the assumption of a bounded past. The definitions of invariant configurations remain the same and the reduction of them to past-time transition configurations is straightforward since no dependency contains both future- and past-time temporal constructs. Furthermore, dependencies containing the temporal operator since (a conventional past-time analogue of the operator until) are not much harder to handle than $\diamondminus$-dependencies. The reduction of an arbitrary $\mathrm{PTL}^{\mathrm{P}}$ formula to an invariant with past time is also straightforward.

13.3 Generalised Conditional Liveness Formulas and Invariants

Conditional liveness formulas and invariants require that any operand of $\Diamond_{\mathrm{m}}$ and $\Diamond$, respectively, is a state formula. We can slightly relax this requirement and permit arbitrary formulas in NL^1. This makes invariants more succinct since a formula such as *sfin* w can now be expressed using only one dependency such as $u_k \equiv \Diamond(\mathit{empty} \wedge w)$ instead of requiring two. The formula $\Box\Diamond^+ w$ can be expressed with the invariant $u_k \equiv \Diamond(w \wedge \bigcirc u_k)$. The overall analysis of such invariants only differs slightly from that for the basic version of invariants. Invariants with *until*-dependencies (see §13.1) can be analogously generalised to permit *until*-dependencies of the form $u_k \equiv (T\,\mathcal{U}\,T')$, where both T and T' are in NL^1.

Transition configurations containing generalised liveness formulas might be of use as a notation for representing deterministic and nondeterministic ω-automata in temporal logic. However, we need to employ *Quantified* PTL (QPTL) to existentially quantify over the variables which collectively encode such an automaton's internal state. Further details of this are omitted here.

13.4 Fusion Logic

Regular expressions are a standard notation for representing regular languages. However, within PITL, it is more appropriate to use languages based on the fusion operator rather than conventional concatenation. This involves 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. This subset will then provide the basis for a generalisation of PTL called *Fusion Logic* (FL) which is also itself a subset of PITL. We originally used Fusion Logic in [57] as a kind of intermediate logic when we reduced the problem of showing axiomatic completeness of Propositional Interval Temporal Logic (PITL) with finite time to showing axiomatic completeness for PTL. Fusion Logic is closely related to Propositional Dynamic Logic (PDL) [21, 22, 30–32, 39]. A major reason for discussing Fusion Logic here is because it is not hard to extend our decision procedure for PTL with finite time to also handle more expressive interval-oriented FL formulas by simply reducing FL formulas to lower level PTL formulas of the kinds already discussed. This demonstrates another link between PTL and intervals and has practical applications.

Definition 86 (Fusion Expression Formulas) *The set of* fusion expression formulas*, denoted* FE*, consists of* PITL *formulas with the syntax given below, where w is a state formula, T is in* NL^1 *and E and F themselves denote* FE *formulas:*

$$w? \qquad E \vee F \qquad \$\,T \qquad E;F \qquad E^* \ .$$

The syntax of FE *formulas is like that of programs in Propositional Dynamic Logic 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, *false*? is permitted even though it is unsatisfiable. Consider the following FE formula:

$$\big((\$\bigcirc p);(q?)\big) \vee (\$\,\neg q)^*.$$

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.

Remark 87 (Expressing concatenation) *It is important to note that the conventional concatenation of two* FE *formulas E and F can be achieved through the use of the* FE *formula $E;(\$\,\mathit{true});F$. Here $\$\,\mathit{true}$ is itself an* FE *formula which is an alternative way to express the* PTL *operator skip. This temporal operation on E and F is sometimes called* "chomp"*, since it is a slight variation of chop. Hence, in the context of temporal logic,* FE *formulas can largely subsume regular expressions although there are slightly different conventions for such things as empty words. We omit the details.*

We now present the sublogic of PITL called here Fusion Logic. In essence, Fusion Logic augments conventional PTL with the fusion expression formulas already introduced.

Definition 88 (Fusion Logic) *Here is the syntax of* FL *where p is any propositional variable, 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*") using the primitive* PITL *constructs chop and* $\neg$*:*

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

Within an FL formula, $\bigcirc$, $\Diamond$ and FL-chop are treated as primitive constructs. Unlike PITL, FL limits the left sides of chop to being FE formulas.

In [57], we described an earlier version of FL having *skip* as a primitive FE formula instead of $\$\,T$. As we noted earlier in Remark 87, the PTL formula *skip* can be expressed in FE as $\$\,true$. The two versions of FL can readily be shown to be equally expressive since $\$\,T$ can be replaced with a semantically equivalent disjunction of formulas by using of ?, *skip* and chop. For example, the FE formula $\$(p \supset \bigcirc q)$ is semantically equivalent to the FE formula $((\neg p)?; skip) \vee (skip; q?)$. In practice, the version described here is much more natural and succinct.

Henriksen and Thiagarajan [33, 34] investigate a formalism related to Wolper's ETL [70, 72] 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 the FE operators ? and \$.

Remark 89 *The temporal operators* $\bigcirc$ *and* $\Diamond$ *which are primitives in* FL *can actually be expressed as instances of* FL*-chop if finite time is assumed:*

$$\models\ \bigcirc X \equiv \langle\$\,true\rangle X \qquad \models\ \Diamond X \equiv \langle(\$\,true)^*\rangle X.$$

In spite of FL being a proper subset of PITL, they have the same expressiveness. This is discussed in [57], where a hierarchical reduction of FL formulas to PTL formulas is also given but is limited to dealing with finite-time intervals. This reduction provides the basis of a decision procedure for FL with finite-time. We plan to describe in future work a hierarchical reduction to transition configurations (also restricted to finite-time). Such transition configurations can then be tested with the decision procedure described in Sect. 8. Like the first reduction in [57], this reduction can also be used for proving the completeness of an axiom system for FL with finite time.

14 Discussion

We conclude with a look at some issues connected with PTL and FL.

As noted earlier, a number of PTL decision procedures are tableau-based algorithms. These include ones described by Wolper [73], Emerson [20] and Lichtenstein and Pnueli [43]. It appears that with some care a tableau-based approach can be hierarchically reduced to our framework. We hope to look into this in more detail in the future.

The BDD-based techniques described in Sect. 8 can be adapted to check in real time that an executing system is not violating assertions expressed in PTL or FL as it runs. Whether FL in particular is useful for this in practice is unclear. In addition, it would appear that the reachability analysis necessary for our approach to work can, as with Bounded Model Checking (BMC) [13], employ SAT-based techniques for PTL and FL instead of BDDs. However, such a SAT-based approach, unlike the BDD-based one, normally cannot exhaustively test for unsatisfiability because in BMC there is no notion corresponding to convergence of BDDs to the set of all atoms reachable from some starting one. Rather BMC works by employing SAT to find at most a single solution not exceeding some predetermined maximum bounded length which for practical reasons is generally much less than the worst-case bounds derived from formula syntax. If a solution is not found, this is typically not by itself sufficient to exclude the existence of larger satisfying intervals.

We have used versions of invariants, transition formulas and conditional liveness formulas to analyse Propositional Dynamic Logic (PDL) without the need for Fischer-Ladner closures. Indeed, this was the original motivation for conditional liveness formulas. However, at present the benefits and novelty of utilising our approach for PDL are less compelling than for PTL.

Acknowledgements

We thank Antonio Cau, Jordan Dimitrov, Rodolfo Gómez and Helge Janicke for comments on versions of this work. In the course of discussions, Howard Bowman, Shmuel Katz, Maciej Koutny and Simon Thompson also made helpful suggestions leading to improvements in the presentation of the material. We are especially grateful to Hussein Zedan for his patience and encouragement during the time this research was undertaken.

References

[1] ANSI. Common Lisp: Standard ANSI INCITS 226-1994 (R1999) (formerly ANSI X3.226-1994 (R1999)). URL: `http://www.ansi.org`, 1999.

[2] B. Banieqbal and H. Barringer. A study of an extended temporal logic and a temporal fixed point calculus. Technical Report UMCS-86-10-2, Dept. of Computer Science, University of Manchester, England, Oct. 1986. revised June 1987.

[3] I. Beer, S. Ben-David, et al. The temporal logic Sugar. In G. Berry, H. Comon, and A. Finkel, editors, *13th Conference on Computer-Aided Verification (CAV01), Paris, France, 18–22 July 2001*, volume 2102 of *LNCS*, pages 363–367, Berlin, 2001. Springer-Verlag.

[4] M. Ben-Ari, Z. Manna, and A. Pnueli. The temporal logic of branching time. In *Eighth ACM Symposium on Principles of Programming Languages*, pages 164–176. ACM, JAN 1981.

[5] M. Ben-Ari, Z. Manna, and A. Pnueli. The temporal logic of branching time. *Acta Informatica*, 20(3):207–226, 1983.

[6] O. Bernholtz, M. Y. Vardi, and P. Wolper. An automata-theoretic approach to branching-time model checking. In *Computer Aided Verification, Proc. 6th Int'l. Workshop*, volume 818 of *LNCS*, pages 142–155, Stanford, California, June 1994. Springer-Verlag.

[7] A. Bolotov, M. Fisher, and C. Dixon. On the relationship between ω-automata and temporal logic normal forms. *Journal of Logic and Computation*, 12(4):561–581, Aug. 2002. Available as `http://www3.oup.co.uk/logcom/hdb/Volume_12/Issue_04/pdf/120561.pdf`.

[8] R. E. Bryant. Graph-based algorithms for Boolean function manipulation. *IEEE Transactions on Computers*, C-35(8), 1986.

[9] R. E. Bryant. Symbolic Boolean manipulation with ordered binary-decision diagrams. *ACM Comput. Surv.*, 24(3):293–318, Sept. 1992.

[10] J. R. Burch, E. M. Clarke, K. L. McMillan, D. L. Dill, and L. J. Hwang. Symbolic model checking: 10^{20} states and beyond. *Inf. and Comp.*, 98(2):142–170, June 1992.

[11] Cadence Design Systems. `http://www.cadence.com/`, URL.

[12] B. F. Chellas. *Modal Logic: An Introduction.* Cambridge University Press, Cambridge, England, 1980.

[13] E. Clarke, A. Biere, R. Raimi, and Y. Zhu. Bounded model checking using satisfiability solving. *Formal Methods in System Design*, 19(1), July 2001.

[14] E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking.* MIT Press, Cambridge, Massachusetts, 2000.

[15] CLISP: An ANSI Common Lisp implementation. `http://clisp.cons.org`, URL.

[16] O. Coudert, C. Berthet, and J. C. Madre. Verification of sequential machines using boolean functional vectors. In L. Claesen, editor, *Proc. IFIP International Workshop on Applied Formal Methods for Correct VLSI Design*, pages 111–128, Leuven, Belgium, Nov. 1989.

[17] O. Coudert, C. Berthet, and J. C. Madre. Verification of synchronous sequential machines based on symbolic execution. In J. Sifakis, editor, *Automatic Verification Methods for Finite State Systems, International Workshop, Grenoble, France, June 12-14, 1989, Proceedings*, volume 407 of *Lecture Notes in Computer Science*, pages 365–373. Springer, 1989.

[18] O. Coudert, C. Berthet, and J. C. Madre. A unified framework for the formal verification of sequential circuits. In *Proc. IEEE International Conf. on Computer Aided Design*, pages 126–129, Nov. 1990.

[19] Colorado University Decision Diagram Package (CUDD). Available at `http://vlsi.colorado.edu/~fabio`, URL.

[20] E. A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B: Formal Models and Semantics, chapter 16, pages 995–1072. Elsevier/MIT Press, Amsterdam, 1990.

[21] M. J. Fischer and R. E. Ladner. Propositional modal logic of programs (extended abstract). In *Conference Record of the Ninth Annual ACM Symposium on Theory of Computing*, pages 286–294, Boulder, Colorado, 2–4 May 1977.

[22] M. J. Fischer and R. E. Ladner. Propositional dynamic logic of regular programs. *J. Comput. Syst. Sci.*, 18(2):194–211, Apr. 1979.

[23] M. Fisher. A normal form for first-order temporal formulae. In D. Kapur, editor, *Automated Deduction - CADE-11, 11th International Conference on Automated Deduction, Saratoga Springs, NY, USA, June 15-18, 1992, Proceedings*, volume 607 of *LNCS*, pages 370–384. Springer-Verlag, 1992.

[24] M. Fisher. A normal form for temporal logic and its application in theorem-proving and execution. *Journal of Logic and Computation*, 7(4):429–456, Aug. 1997.

[25] M. Fisher, C. Dixon, and M. Peim. Clausal temporal resolution. *ACM Transactions on Computational Logic*, 2(1):12–56, Jan. 2001.

[26] T. French. A proof of the completeness of PLTL. Available as `http://www.cs.uwa.edu.au/~tim/papers/pltlcomp.ps`, 2000.

[27] D. Gabbay, A. Pnueli, S. Shelah, and J. Stavi. On the temporal analysis of fairness. In *Seventh Annual ACM Symposium on Principles of Programming Languages*, pages 163–173, 1980.

[28] R. Goldblatt. *Logics of Time and Computation*, volume 7 of *CSLI Lecture Notes*. CLSI/SRI International, 333 Ravenswood Av., Menlo Park, CA 94025, 1987.

[29] J. Halpern, Z. Manna, and B. Moszkowski. A hardware semantics based on temporal intervals. In J. Diaz, editor, *Proceedings of the 10-th International Colloquium on Automata, Languages and Programming*, volume 154 of *LNCS*, pages 278–291, Berlin, 1983. Springer-Verlag.

[30] D. Harel. Dynamic logic. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume II, pages 497–604. Reidel Publishing Company, Dordrecht, 1984.

[31] D. Harel, D. Kozen, and J. Tiuryn. *Dynamic Logic*. MIT Press, Cambridge, Massachusetts, 2000.

[32] D. Harel, D. Kozen, and J. Tiuryn. Dynamic logic. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume 4, pages 99–217. Kluwer Academic Publishers, Dordrecht, 2nd edition edition, 2002.

[33] J. G. Henriksen and P. S. Thiagarajan. Dynamic linear time temporal logic. Technical Report RS-97-8, BRICS, Department of Computer Science, University of Aarhus, Aarhus, Denmark, Apr. 1997. Available at `http://www.brics.dk/RS/97/8/`.

[34] J. G. Henriksen and P. S. Thiagarajan. Dynamic linear time temporal logic. *Annals of Pure and Applied Logic*, 96(1-3):187–207, 1999.

[35] Y. Hollander, M. Morley, and A. Noy. The *e* language: A fresh separation of concerns. In *Technology of Object-Oriented Languages and Systems (Proceedings of 38th Int'l. TOOLS Conference, TOOLS Europe 2001)*, pages 41–50. IEEE Computer Society Press, Mar. 2001. All authors at Verisity, Ltd., Rosh-Ha-Ain, Israel. Presented at TOOLS Europe 2001, 38th International TOOLS Conference, Zürich Technopark, Zürich, Switzerland, March 12–14, 2001.

[36] G. E. Hughes and M. J. Cresswell. *A New Introduction to Modal Logic*. Routledge, London, 1996.

[37] IEEE Candidate Standard 1647. Produced by the **e** Functional Verification Language Working Group. `http://www.ieee1647.org/`, URL.

[38] Interval Temporal Logic (ITL) homepage. `http://www.cse.dmu.ac.uk/STRL/ITL/`, URL.

[39] D. Kozen and J. Tiuryn. Logics of programs. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 789–840. Elsevier Science Publishers, Amsterdam, 1990.

[40] F. Kröger. *Temporal Logic of Programs*, volume 8 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, 1987.

[41] T. Kropf. *Introduction to Formal Hardware Verification*. Springer-Verlag, Heidelberg, Germany, 1999.

[42] M. Lange and C. Stirling. Focus games for satisfiability and completeness of temporal logic. In *Proc. 16th Annual IEEE Symp. on Logic in Computer Science, LICS'01*, pages 357–365, Boston, MA, USA, June 2001. IEEE Computer Society Press.

[43] O. Lichtenstein and A. Pnueli. Propositional temporal logics: Decidability and completeness. *Logic Journal of the IGPL*, 8(1):55–85, 2000. Available at `http://www3.oup.co.uk/igpl/Volume_08/Issue_01/#Lichtenstein`.

[44] Z. Manna and A. Pnueli. Verification of concurrent programs: the temporal framework. In R. S. Boyer and J. S. Moore, editors, *The Correctness Problem in Computer Science*, pages 215–273, New York, 1981. Academic Press.

[45] Z. Manna and A. Pnueli. The anchored version of the temporal framework. In J. W. D. Bakker, W.-P. de Roever, and G. Rozenberg, editors, *Linear Time, Branching Time, and Partial Order in Logics and Models for Concurrency (REX Workshop 1988)*, volume 354 of *LNCS*, pages 201–284. Springer-Verlag, 1989.

[46] K. L. McMillan. *Symbolic model checking*. Kluwer Academic Publishers, Boston, Mass., 1993.

[47] M. J. Morley. Semantics of temporal ***e***. In T. F. Melham and F. G. Moller, editors, Banff'99 *Higher Order Workshop: Formal Methods in Computation, Ullapool, Scotland, 9–11 Sept. 1999*, pages 138–142. University of Glasgow, Department of Computing Science Technical Report, 1999.

[48] B. Moszkowski. *Reasoning about Digital Circuits*. PhD thesis, Department of Computer Science, Stanford University, June 1983. Technical report STAN–CS–83–970.

[49] B. Moszkowski. A temporal logic for multi-level reasoning about hardware. In *Proceedings of the 6-th Interna-*

tional Symposium on Computer Hardware Description Languages, pages 79–90, Pittsburgh, Pennsylvania, May 1983. North-Holland Pub. Co.

[50] B. Moszkowski. A temporal logic for multilevel reasoning about hardware. *Computer*, 18:10–19, 1985.

[51] B. Moszkowski. *Executing Temporal Logic Programs.* Cambridge University Press, Cambridge, England, 1986.

[52] B. Moszkowski. Some very compositional temporal properties. In E.-R. Olderog, editor, *Programming Concepts, Methods and Calculi*, volume A-56 of *IFIP Transactions*, pages 307–326. IFIP, Elsevier Science B.V. (North–Holland), 1994.

[53] B. Moszkowski. Compositional reasoning about projected and infinite time. In *Proceedings of the First IEEE Int'l Conf. on Engineering of Complex Computer Systems (ICECCS'95)*, pages 238–245. IEEE Computer Society Press, 1995.

[54] B. Moszkowski. Using temporal fixpoints to compositionally reason about liveness. In He Jifeng, J. Cooke, and P. Wallis, editors, *BCS-FACS 7th Refinement Workshop*, electronic Workshops in Computing, London, 1996. BCS-FACS, Springer-Verlag and British Computer Society.

[55] B. Moszkowski. Compositional reasoning using Interval Temporal Logic and Tempura. In W.-P. de Roever, H. Langmaack, and A. Pnueli, editors, *Compositionality: The Significant Difference*, volume 1536 of *LNCS*, pages 439–464, Berlin, 1998. Springer-Verlag.

[56] B. Moszkowski. An automata-theoretic completeness proof for Interval Temporal Logic (extended abstract). In U. Montanari, J. Rolim, and E. Welzl, editors, *Proceedings of the 27th International Colloquium on Automata, Languages and Programming (ICALP 2000)*, volume 1853 of *LNCS*, pages 223–234, Geneva, Switzerland, July 2000. Springer-Verlag.

[57] B. Moszkowski. A hierarchical completeness proof for Propositional Interval Temporal Logic with finite time. *Journal of Applied Non-Classical Logics*, 14(1–2):55–104, 2004. Special issue on Interval Temporal Logics and Duration Calculi. V. Goranko and A. Montanari guest eds.

[58] B. Moszkowski. A hierarchical completeness proof for propositional temporal logic. In N. Dershowitz, editor, *Verification: Theory and Practice: Essays Dedicated to Zohar Manna on the Occasion of His 64th Birthday*, volume 2772 of *LNCS*, pages 480–523. Springer-Verlag, Heidelberg, 2004.

[59] The Perl programming language. `http://www.perl.org`, URL.

[60] PerlDD: Perl extensions to CUDD [19]. Available at `http://vlsi.colorado.edu/~fabio`, URL.

[61] A. Pnueli. The temporal logic of programs. In *Proceedings of the 18th Symposium on the Foundation of Computer Science*, pages 46–57. ACM, 1977.

[62] V. R. Pratt. Process logic. In *Sixth Annual ACM Symposium on Principles of Programming Languages*, pages 93–100, 1979.

[63] PSL/Sugar Consortium. `http://www.pslsugar.org`, URL.

[64] R. Pucella. Logic column 11: The finite and the infinite in temporal logic. *SIGACT News*, 36(1):86–99, 2005, cs.LO/0502031. Available at Computing Research Repository (CoRR): `http://arxiv.org/abs/cs.LO/0502031`.

[65] N. Rescher and A. Urquhart. *Temporal Logic*. Springer-Verlag, New York, 1971.

[66] Systemverilog website. `http://www.systemverilog.org`, URL.

[67] M. Y. Vardi and P. Wolper. Automata-theoretic techniques for modal logics of programs. *J. Comput. Syst. Sci.*, 32(2):183–221, Apr. 1986.

[68] Verisity Ltd. (acquired by Cadence Design Systems [11] in 2005). `http://www.cadence.com/verisity/`, URL.

[69] Verisity Ltd. Semantics of temporal ***e***. Revised version of Morley [47]. Available from website of IEEE candidate standard 1647 as `http://www.ieee1647.org/downloads/temporale_denotational.pdf`, Dec. 2003.

[70] P. Wolper. Temporal logic can be more expressive. In *Proc. 22nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 340–348, Nashville, Tennessee, Oct. 1981. IEEE Computer Society.

[71] P. Wolper. Constructing automata from temporal logic formulas: A tutorial. In *Lectures on Formal Methods in Performance Analysis (First EEF/Euro Summer School on Trends in Computer Science)*, volume 2090 of *LNCS*, pages 261–277. Springer-Verlag, July 2001.

[72] P. L. Wolper. Temporal logic can be more expressive. *Information and Control*, 56(1-2):72–99, 1983.

[73] P. L. Wolper. The tableau method for temporal logic: An overview. *Logique et Analyse*, 110–111:119–136, 1985.