Interconnections between classes of sequentially compositional temporal formulas

Ben Moszkowski

Software Technology Research Laboratory, De Montfort University, Leicester, UK

Abstract

Interval Temporal Logic (ITL) is an established formalism for reasoning about time periods. We elucidate here the relationship between various kinds of compositional propositional ITL formulas. Several are closed under conjunction and the standard temporal operator known as "box" and "always".

Keywords: Formal methods ,Interval Temporal Logic ,Compositionality

1 Introduction

Intervals and discrete linear state sequences offer a compellingly natural and flexible way to model computational processes involving hardware or software. **Interval Temporal Logic (ITL)** [1], an established formalism for reasoning about such phenomena, has operators for sequentially combining formulas. If A and B are formulas, so are $A; B$ (*"chop"*) and A^* (*"chop-star"*). These are related to the concatenation and Kleene star operators for regular expressions. Time is modelled as in conventional discrete linear-time temporal logic using finite and infinite sequences of one or more states.

We recently introduced and investigated **2-to-1** formulas for compositional reasoning [2]. A formula A is 2-to-1 if the implication $(A; A) \supset A$ is valid, so if two portions of a system ensure such a formula's behaviour, then their sequential composition is guaranteed to as well. This work builds on our earlier compositional techniques surveyed in [3] to facilitate inference rules for combining concurrent systems sequentially and in parallel. A sample rule is later presented in Sect. 3. The 2-to-1 formulas are closed under conjunction and the conventional temporal operator $\Box$ ("always") which examines *suffix* subintervals. This helps modularly obtain 2-to-1 safety and liveness formulas such as the standard temporal formulas $\Box p$ ("always p") and $\Box(p \supset \Diamond q)$ ("p always leads to q"), where p and q are propositional variables. The propositional version of ITL (**PITL**) used

here to formalise 2-to-1 formulas is decidable and has a complete axiomatisation [4], but our results are also applicable to first-order ITL (along the lines of [3]).

The versatile class of 2-to-1 formulas has useful subclasses (e.g., $*$**-to-1**: $A^* \supset A$) and other variants, also closed under conjunction and sometimes closed under $\Box$ as well. Our presentation here mainly concerns interconnections between these to elucidate the nature of such formulas and obtain new ones. We systematically and incrementally apply properties proved about some formulas to later proofs for others to avoid redundant reasoning.

2 Propositional Interval Temporal Logic

We now describe the version of (quantifier-free) PITL used here. More about PITL can be found in [4] (see also [1] and the ITL web pages [5]).

Here is a BNF syntax of PITL formulas, with p any propositional variable:

$$A ::= \mathit{true} \mid p \mid \neg A \mid A \vee A \mid \mathit{skip} \mid A; A \mid A^*. \tag{1}$$

Boolean operators *false*, $A \wedge B$, $A \supset B$ and $A \equiv B$ are defined as usual.

Time within PITL is modelled by discrete, linear state sequences. The **set of states** Σ is the power set 2^V, where V is the set of propositional variables. Each state in Σ therefore sets every propositional variable $p, q, \ldots$ to *true* or *false*. The associated (standard) version of PITL with such state-based variables (instead of interval-based ones) is called **local PITL**. An **interval** σ is any element of $\Sigma^+ \cup \Sigma^\omega$ and has states σ^0, σ^1, ...

The notation $\sigma \models A$, defined shortly by induction on formula A's syntax, denotes that interval σ **satisfies** A. If all intervals satisfy A, denoted $\models A$, it is **valid**. Below are the semantics of the first five PITL constructs in (1):

$$\sigma \models \mathit{true} \text{ for any } \sigma \qquad \sigma \models p \text{ iff } p \in \sigma^0 \text{ (initially } p) \qquad \sigma \models \neg A \text{ iff } \sigma \not\models A$$
$$\sigma \models A \vee B \text{ iff } \sigma \models A \text{ or } \sigma \models B \qquad \sigma \models \mathit{skip} \text{ iff } \sigma \in \Sigma^2 \text{ (two states).}$$

The cases below for chop and chop-star involve **subintervals**:

- Chop: $\sigma \models A; B$ iff for some σ' and σ'', $\sigma' \models A$ and $\sigma'' \models B$
  or $\sigma \in \Sigma^\omega$ and $\sigma \models A$,

  where $\sigma' \in \Sigma^+$ is a finite **prefix** subinterval of σ (perhaps even σ itself if $\sigma \in \Sigma^+$), and σ'' is the adjacent **suffix** subinterval of σ with one shared state (i.e., the last state of σ'). Chop here is **weak** (like the weak version of the temporal operator Until) for potentially nonterminating programs which ignore B. **Strong** chop (and chop-star) is derivable.

- Chop-star: $\sigma \models A^*$ iff one of the following holds: **(1)** σ has only one state (i.e., $\sigma \in \Sigma$). **(2)** σ either itself satisfies A or can be split into a finite number of subintervals which share end-states (like chop) and all satisfy A. **(3)** $\sigma \in \Sigma^\omega$ and can be split into ω finite-length intervals sharing end-states (like chop) and each satisfying A.

$\bigcirc A \mathrel{\widehat{=}} \mathit{skip}; A$	Next	$\mathit{more} \mathrel{\widehat{=}} \bigcirc \mathit{true}$	Two or more states
$\mathit{empty} \mathrel{\widehat{=}} \neg \mathit{more}$	One state	$A? \mathrel{\widehat{=}} \mathit{empty} \wedge A$	One state with test
$\mathit{inf} \mathrel{\widehat{=}} \mathit{true};\mathit{false}$	ω states	$\mathit{finite} \mathrel{\widehat{=}} \neg \mathit{inf}$	Finite interval
$\Diamond A \mathrel{\widehat{=}} \mathit{finite}; A$	Eventually	$\Box A \mathrel{\widehat{=}} \neg\Diamond\neg A$	Henceforth (always)
$\mathit{fin}\, A \mathrel{\widehat{=}} \Box(\mathit{empty} \supset A)$	Final state (weak)	$A^\omega \mathrel{\widehat{=}} \mathit{inf} \wedge (\mathit{finite} \wedge A)^*$	Chop-omega
$\diamondsuit A \mathrel{\widehat{=}} A; \mathit{true}$	Initial (prefix) subinterval	$\boxdot A \mathrel{\widehat{=}} \neg \diamondsuit \neg A$	All initial (prefix) subintervals

Table 1: Some Useful Derived PITL Operators

Consider a sample 5-state interval σ with the following alternating values for the variable p:

$$p \quad \neg p \quad p \quad \neg p \quad p.$$

Here are formulas satisfied by σ:

$$p \quad \mathit{skip}; \neg p \quad p \wedge (\mathit{true}; \neg p) \quad (p \wedge (\mathit{skip}; \mathit{skip}))^*.$$

For instance, $\mathit{skip}; \neg p$ is true since σ's prefix subinterval $\sigma^0\sigma^1$ satisfies skip and the adjacent suffix subinterval $\sigma^1 \ldots \sigma^4$ satisfies $\neg p$ because $p \notin \sigma^1$. The formula $(p \wedge (\mathit{skip}; \mathit{skip}))^*$ is true since σ's subintervals $\sigma^0\sigma^1\sigma^2$ and $\sigma^2\sigma^3\sigma^4$ both satisfy $p \wedge (\mathit{skip}; \mathit{skip})$. The interval σ does not satisfy the formulas $\neg p$, $\mathit{skip}; p$ and $\mathit{true}; (\neg p \wedge \neg(\mathit{true}; p))$.

Table 1 shows useful derived PITL operators.

Let w and w' denote **state formulas** with no temporal operators. Let **PTL** denote the PITL subset of conventional **Propositional Linear-Time Temporal Logic** with just the (derived) operators $\bigcirc$ and $\Diamond$ in Table 1.

Here are some sample valid PITL formulas:

$$A \supset \diamondsuit A \quad \mathit{skip}^* \quad \mathit{inf} \equiv \Box \mathit{more} \quad (w \wedge A); B \equiv w \wedge (A; B) \quad A \equiv (\mathit{empty}; A).$$

As another example, the equivalence $(w \wedge A) \equiv (\mathit{empty} \wedge w); A$ is valid since for any interval σ, $\sigma \models w \wedge A$ iff σ's first state satisfies w and $\sigma \models A$.

3 2-to-1 Formulas

A PITL formula A is **2-to-1** iff $(A; A) \supset A$ is valid. For example, true, p, empty and $B?$ (for any B) are 2-to-1, as are PTL formulas $\Box p$ and $\Box(p \supset \Diamond q)$, but not skip. The next sample semantic inference rule uses a 2-to-1 formula A with systems Sys and Sys' and pre- and post-conditions w, w' and w'':

$$\frac{\models\ w \wedge \mathit{Sys}\ \supset\ A \wedge \mathit{fin}\ w', \quad \models\ w' \wedge \mathit{Sys}'\ \supset\ A \wedge \mathit{fin}\ w''}{\models\ w \wedge (\mathit{Sys}; \mathit{Sys}')\ \supset\ A \wedge \mathit{fin}\ w''.}$$

We now discuss properties of 2-to-1 formulas. This class is later used with other classes such as $*$**-to-1** formulas (i.e., $\models A^* \supset A$) for iteration in Sect. 4. Our systematic analysis incrementally obtains formulas in the classes. Included are many PTL formulas for which it also seems computationally feasible to check membership in the classes, but this is left for future work.

Theorem 1 (Moszkowski [2])**.** *2-to-1 formulas are closed under* $\wedge$ *and* $\Box$*.*

Proof. Here is a chain of valid implications for $\wedge$: $(A \wedge B);(A \wedge B) \supset (A;A) \wedge (B;B) \supset (A \wedge B)$. For $\Box$, if $\sigma \models (\Box A);(\Box A)$, then every suffix of σ satisfies $A;A$ or A and hence A so $\sigma \models \Box A$. Therefore, $\models (\Box A);(\Box A) \supset \Box A$. $\Box$

The $\Box$-closure for 2-to-1 formulas nicely generalises to the next semantic inference rule for any C and C': $\models (C';C) \supset C \;\Rightarrow\; \models ((\Box C');\Box C) \supset \Box C$.

If A is 2-to-1, so is $\mathit{empty} \vee A$ (equivalent to $\mathit{more} \supset A$) and, more generally, $B? \vee A$ (empty is equivalent to $\mathit{true}?$). Our proof reduces $(B? \vee A);(B? \vee A)$ to $(B?;(B? \vee A)) \vee (A;(B?)) \vee (A;A)$ and then $B? \vee A$ using, e.g., $\models (A;(B?)) \supset A$. The 2-to-1 class is *not* closed under $\vee$ (e.g., $p \vee \Box q$).

We define a PITL formula A to be $\diamondsuit$**-to-1** if $\models (\diamondsuit A) \supset A$ is valid. This includes 2-to-1 formulas p, $\Diamond q$ and $p \supset \Diamond q$, but not $\Box p$ so lacks $\Box$-closure.

Lemma 2. *Every* $\diamondsuit$*-to-1 formula is 2-to-1.*

Proof. From $\models (A;A) \supset \diamondsuit A$ and $\models (\diamondsuit A) \supset A$ follows $\models (A;A) \supset A$. $\Box$

Lemma 3. *The* $\diamondsuit$*-to-1 formulas are closed under both* $\wedge$ *and* $\vee$*.*

Proof. Here are chains of valid implications:

$$\diamondsuit(A \wedge B) \;\supset\; (\diamondsuit A) \wedge (\diamondsuit B) \;\supset\; A \wedge B$$
$$\diamondsuit(A \vee B) \;\supset\; (\diamondsuit A) \vee (\diamondsuit B) \;\supset\; A \vee B.$$

$\Box$

Proofs for Theorem 1 and Lemmas 2–3 work even if chop is the only primitive temporal operator, time is not discrete, or variables $p, q, \ldots$ are not state- but interval-based (e.g., σ^0 can set p true, and $\sigma^0\sigma^1$ can set p false).

The $\diamondsuit$-to-1 formulas such as those in Lemma 4 below can serve as simple 2-to-1 building blocks (recall Lemma 2) for creating bigger 2-to-1 formulas.

Lemma 4. *For any state formula* w*, PITL formula* A *and* $\diamondsuit$*-to-1 formula* B*, the following are* $\diamondsuit$*-to-1:* w*,* $A;B$ *(e.g.,* $\diamondsuit A$*,* $\bigcirc B$ *and* $\Diamond B$*) and* $w \supset B$*.*

Proof. For example, here is a chain of valid implications for $A;B$: $\diamondsuit(A;B) \supset A;\diamondsuit B \supset A;B$. We re-express $w \supset B$ as $\neg w \vee B$ and use closure under $\vee$. $\Box$

For instance, the PTL formula $p \supset \Diamond q$ is $\diamondsuit$-to-1 and so 2-to-1 as is the liveness formula $\Box(p \supset \Diamond q)$ by $\Box$-closure. However, this is not $\diamondsuit$-to-1.

Let a **positive** PTL formula be one built from state formulas, $\bigcirc$ and $\Diamond$ using conjunctions and disjunctions, but no negated temporal formulas.

Theorem 5. *Every positive PTL formula is $\diamond$-to-1 and hence also 2-to-1.*

Proof. We use Lemmas 3 and 4 and induction on syntax, then Lemma 2. □

Let **NL**^{1} (NL="**Next Logic**") be the $\diamond$-less PTL subset with no $\bigcirc$ nested in another $\bigcirc$ (e.g., $p \wedge \neg q$, $(\bigcirc p) \supset \neg \bigcirc (q \wedge \neg r)$ and $p \equiv \bigcirc \neg p$, but not $p \wedge \bigcirc \bigcirc q$). Let T denote an NL^1 formula. It sees at most an interval's first two states, just as a state formula such as $p \wedge \neg q$ only sees the interval's first state. Positive NL^1 formulas are $\diamond$-to-1 by Theorem 5, but not all NL^1 formulas are (e.g., $\not\models (\diamond \mathit{empty}) \supset \mathit{empty}$). We however show they are 2-to-1.

Lemma 6. *For any NL^1 formula T, $\mathit{more} \wedge T$ is $\diamond$-to-1 and T is 2-to-1.*

Proof. We can re-express $\mathit{more} \wedge T$ as a positive NL^1 formula (e.g., $\mathit{more} \wedge \neg(p \wedge \bigcirc q)$ becomes $\mathit{more} \wedge (\neg p \vee \bigcirc \neg q)$), so it is $\diamond$-to-1 by Theorem 5, and also 2-to-1 (Lemma 2). Let us re-express T as $(\mathit{empty} \wedge T) \vee (\mathit{more} \wedge T)$. This abbreviates to $T? \vee (\mathit{more} \wedge T)$ and is an instance of the 2-to-1 disjunction $B? \vee A$ discussed earlier after Theorem 1, so T is 2-to-1. □

For example, the NL^1 formula $\mathit{more} \supset (p \equiv \bigcirc p)$ tests equality of p in the first two states. It is 2-to-1 by Lemma 6. By $\Box$-closure, so is $\Box(\mathit{more} \supset (p \equiv \bigcirc p))$. This formula, denoted ***stable p***, tests that p is stable in the interval.

A restricted **until** formula T *until* C is definable as $(\mathit{finite} \wedge \Box(\mathit{more} \supset T)); C$, also expressible as $\big(\mathit{finite} \wedge (\mathit{skip} \wedge T)^*\big); C$. Here is an example: $(p \equiv \bigcirc p)$ *until* $\Box q$. We omit *finite* for a *weak* version. If C is $\diamond$-to-1, so are T *until* C and its weak variant by Lemma 4, and they are also 2-to-1 by Lemma 2. We further illustrate T *until* C with a variant of a formula Bäumler et al. [6] use with ITL for Jones' rely-guarantee framework [7]: G *until* $(\mathit{more} \wedge G \wedge \neg R)$, where G and R are in NL^1. This ensures G is true longer than R. Here C is the NL^1 formula $\mathit{more} \wedge G \wedge \neg R$ and is $\diamond$-to-1 by Lemma 6. Hence, our instance of T *until* C is indeed $\diamond$-to-1 and 2-to-1 (as is one using the weak version of *until*). Weak variants of it using disjunctions with either $\Box G$ (see Bäumler et al. [6]), $\mathit{inf} \wedge \Box G$ or $\Box(\mathit{more} \supset G)$ are 2-to-1. The proofs use the fact that if A is 2-to-1, B is $\diamond$-to-1 and $\models (A;B) \supset B$, then $A \vee B$ is 2-to-1. This is so since we can express $(A \vee B); (A \vee B)$ as $(A;A) \vee (A;B) \vee \big(B; (A \vee B)\big)$ and reduce it to $A \vee B$. Here $B; (A \vee B)$ implies $\diamond B$, so also B since B is $\diamond$-to-1.

4 Iteration and Star-to-1 Formulas

The **$*$-to-1** class (i.e., $\models A^* \supset A$) includes $\mathit{more} \supset B$ and $\Box(\mathit{more} \supset B)$ for any B in $\diamond$-to-1 or NL^1 (e.g., p and $p \supset \diamond q$). We analyse it using the variants below:

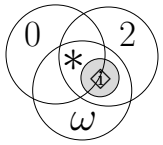
0-to-1 formulas: $\models \mathit{empty} \supset A$

ω-to-1 formulas: $\models A^\omega \supset A$.

Classes 0-to-1, 2-to-1 and ω-to-1 do not contain each other, but their intersection is $*$-to-1 (Lemma 7). Below are sample formulas and their classes:

$$\mathit{skip}\ [\,] \qquad \mathit{empty} \vee \mathit{skip}\ [0] \qquad \mathit{finite} \wedge p\ [2] \qquad \mathit{finite}\ [0,2]$$
$$\mathit{skip} \vee \mathit{inf}\ [\omega] \qquad \neg(\mathit{skip};\mathit{skip})\ [0,\omega] \qquad \Box p\ [2,\omega]$$
$$\mathit{more} \supset p\ [0,2,\omega,*] \qquad p\ [\diamondsuit,2,\omega] \qquad \mathit{true}\ [\diamondsuit,0,2,\omega,*],$$

where the sixth formula $\neg(\mathit{skip};\mathit{skip})$ denotes "not three states". The intersection of $\diamondsuit$-to-1 and 0-to-1 is all valid formulas (***true*-to-1**). The next Venn diagram's shaded area is for $\diamondsuit$-to-1:



($\diamondsuit$-to-1 is a subset of 2-to-1 and ω-to-1 by Lemmas 2 and 10.)

Lemma 7. *A formula A is $*$-to-1 iff it is 0-to-1, 2-to-1 and ω-to-1.*

Proof. Left to right: The formulas *empty*, $A;A$ and A^ω each imply A^*. Hence, for example, $\models A^\omega \supset A^*$ and assumption $\models A^* \supset A$ yield that A is ω-to-1.

Right to left: Assume $\sigma \models A^*$. Chop-star's semantics in Sect. 2 has three cases: either $\sigma \models \mathit{empty}$, $\sigma \models A^n$, or $\sigma \models A^\omega$, where A^n denotes n iterations for some (finite) $n \geq 1$ (e.g., $A;A;A$). If $\sigma \models \mathit{empty}$, then $\sigma \models A$ since A is 0-to-1. The assumption that A is 2-to-1 and induction on any $k \geq 1$ yield $\models A^k \supset A$, so if $\sigma \models A^n$, then $\sigma \models A$. Lastly, if $\sigma \models A^\omega$, then $\sigma \models A$ since A is ω-to-1. Thus, for each case of $\sigma \models A^*$, also $\sigma \models A$. So $\models A^* \supset A$. □

Lemma 8. *If sets of formulas $S_1, \ldots, S_n$ are each closed under a PITL operator (e.g., $\wedge$ or $\Box$), so is their intersection $S_1 \cap \cdots \cap S_n$.*

Proof. For example with $\Box$, if $A \in S_1 \cap S_2$, then $\Box A$ is in S_1 and S_2, so in $S_1 \cap S_2$. □

Lemma 9. *The 0-to-1 formulas are closed under $\wedge$, $\vee$ and $\Box$.*

Proof. For example, from $\models \mathit{empty} \supset A$, we readily have $\models \mathit{empty} \supset \Box A$. □

Lemma 10. *The ω-to-1 class includes $\diamondsuit$-to-1 and is closed under $\wedge$ and $\Box$.*

Proof. We use the two chains of valid implications $A^\omega \supset \diamondsuit A \supset A$ (where A is $\diamondsuit$-to-1) and $(A \wedge B)^\omega \supset (A^\omega \wedge B^\omega) \supset (A \wedge B)$. For $\Box$-closure, if $\sigma \models (\Box A)^\omega$, each suffix satisfies A^ω and hence A, so $\sigma \models \Box A$ and $\models (\Box A)^\omega \supset \Box A$. □

Closure for $*$-to-1 under $\wedge$ and $\Box$ follows by Theorem 1 and Lemmas 7–10. The next theorem helps show $\Box\big(\mathit{more} \supset (p \supset \Diamond q)\big)$ and *stable* p are $*$-to-1:

Theorem 11. *If B is $\diamondsuit$-to-1, then $\mathit{more} \supset B$ and $\Box(\mathit{more} \supset B)$ are $*$-to-1.*

Proof. Formula $\mathit{more} \supset B$ is trivially 0-to-1. It is 2-to-1 since B is (see Lemma 2 and the discussion after Theorem 1). Class ω-to-1 includes $\diamondsuit$-to-1 (Lemma 10), so $\models B^\omega \supset B$. This yields $\models B^\omega \supset (\mathit{more} \supset B)$. We re-express B^ω as $(\mathit{more} \supset B)^\omega$, so $\mathit{more} \supset B$ is also ω-to-1, hence $*$-to-1 by Lemma 7. Also, $\Box(\mathit{more} \supset B)$ is $*$-to-1 by $\Box$-closure (so 0-to-1, 2-to-1 and ω-to-1). □

Every positive PTL formula, e.g., $p \supset \Diamond q$, can serve as B (Theorem 5). For any NL[1] formula T, the formula $more \wedge T$ is $\Diamond$-to-1 by Lemma 6. Hence, $more \supset (more \wedge T)$ is $*$-to-1, and so is the equivalent formula $more \supset T$.

Both ω-to-1 and $*$-to-1 are *not* $\vee$-closed (e.g., $(more \supset p) \vee \Box(more \supset q)$).

Variants of Lemmas 2 and 3 about $\Diamond$-to-1 formulas work for **$\Diamond$-to-1** formulas (e.g., $\Diamond A$ and *finite*). For example, $\Diamond$-to-1 is in 2-to-1. In [2], time symmetry yields more formulas. Note that safety properties, which concern prefix subintervals, are expressible with $\boxdot$. For example, $\Box p$ and $\Box(more \supset T)$ are equivalent to $\boxdot \Box p$ and $\boxdot \Box(more \supset T)$, and also equivalent to $\boxdot(finite \supset \Box p)$ and $\boxdot(finite \supset \Box(more \supset T))$, respectively.

References

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

[2] B. Moszkowski, Compositional reasoning using intervals and time reversal, in: 18th Int'l Symp. on Temporal Representation and Reasoning (TIME 2011), IEEE Computer Society, 2011, pp. 107–114.

[3] B. Moszkowski, Compositional reasoning using Interval Temporal Logic and Tempura, in: Compositionality: The Significant Difference, Vol. 1536 of LNCS, Springer-Verlag, Berlin, 1998, pp. 439–464.

[4] B. Moszkowski, A complete axiom system for propositional Interval Temporal Logic with infinite time, Logical Methods in Computer Science 8 (3:10) (2012) 1–56.

[5] ITL web pages, `http://www.tech.dmu.ac.uk/STRL/ITL/`.

[6] S. Bäumler, G. Schellhorn, B. Tofan, W. Reif, Proving linearizability with temporal logic, Formal Aspects of Computing 23 (1) (2011) 91–112.

[7] C. B. Jones, Tentative steps toward a development method for interfering programs, ACM Transactions on Programming Languages and Systems 5 (4) (1983) 596–619.

This author-produced version was formatted on 25 July 2013.