The Workshops of the Thirtieth AAAI Conference on Artificial Intelligence
Planning for Hybrid Systems: Technical Report WS-16-12

Mixed Propositional Metric Temporal Logic: A New Formalism for Temporal Planning

Son Thanh To[1], Mark Roberts[2], Thomas Apker[3], Benjamin Johnson[2], & David W. Aha[3]
[1]Knexus Research Corporation, Springfield, VA 22153
[2]NRC Postdoctoral Fellow; Naval Research Laboratory, Code 5514; Washington, DC 20375
[3]Navy Center for Applied Research in AI; Naval Research Laboratory, Code 5514; Washington, DC 20375
son.to@knexusresearch.com | {mark.roberts.ctr; thomas.apker; benjamin.johnson.ctr; david.aha}@nrl.navy.mil

Abstract

Temporal logics have been used in autonomous planning to represent and reason about temporal planning problems. However, such techniques have typically been restricted to either (1) representing actions, events, and goals with temporal properties or (2) planning for temporally-extended goals under restrictive conditions of classical planning. We introduce Mixed Propositional Metric Temporal Logic (MPMTL), where formulae in MPMTL are built over mixed binary and continuous real variables. MPMTL provides a natural, flexible formalism for representing and reasoning about temporal problems. We analyze the complexity of MPMTL formulae satisfiability and model checking, and identify MPMTL fragments with lower complexity. We also introduce an approach to world modeling using a timeline vector, relevant to temporal planning with continuous change (as opposed to the use of discrete states). Our model supports retroactive action progression, concurrent and overlapping actions with discrete and continuous changes, and concurrent effects to the same variable. For reasoning about this temporal planning problem, we define a progression function for actions with the new temporal properties and a solution to this temporal task.

Introduction

Temporal planning has attracted the attention of researchers in the robotics and AI communities. Numerous systems have advanced temporal planning (Benton, Coles, and Coles 2012; Coles *et al.* 2009; 2010; 2008; Della Penna *et al.* 2009; Do and Kambhampati 2003; Dvorak et al. 2014; Molineaux, Klenk, and Aha 2010; Penberthy and Weld 1994). However, these systems typically consider limited temporal properties. For example, they may assume that action conditions and effects exist at only the beginning, the end, or over the entire duration of an action, but not some other point or interval. Furthermore, the temporal goals they might support are limited to goals with a deadline.

Other approaches (Bacchus and Kabanza 2000; 1998; Bauer and Haslum 2010; Kabanza and Thiebaux 2005; Patrizi, Lipoveztky, and Geffner 2013; Patrizi *et al.* 2011) extend classical planning with temporally extended goals, typically expressed in some temporal logic formalism (e.g., Linear Temporal Logic), while assuming other properties of the problem are restricted to conditions of classical planning (e.g., no durative actions, no temporal conditions or effects).

Several researchers have proposed various temporal logics as possible formalisms for the problem (Allen 1994; Allen and Ferguson 1994). In this paper, we develop *Mixed Propositional Metric Temporal Logic (MPMTL)*, a hybrid of a satisfiability modulo theory (SMT) (Barrett *et al.* 2009; Cimatti *et al.* 2014) and a metric temporal logic (Alur and Henzinger 1989; Koymans 1990). Formulae in MPMTL are built over *mixed* binary and continuous real variables with *metric* modalities. (Although MPMTL can be extended to multi-valued discrete variables, we restrict our attention to binary variables here.) To our knowledge, such a hybrid formalism has not been previously proposed, despite being essential for temporal planning with continuous change. We show that satisfiability and model checking for a normalized form of MPMTL is polynomial. We present a transformation of an MPMTL formula to this normalized form and report on the upper bound of its complexity and the size of a resulting formula. This allows for more expressive fragments of MPMTL that have reasonable computational complexity for temporal planning.

Using MPMTL, we propose a model for temporal planning, where action conditions and effects, constraints, and goals are expressed by a formula in its fragments. We model the world as a *timeline vector*, which is relevant to temporal planning with continuous change. This also enables representing an initial changing world, essential for online planning and re-planning in a dynamic environment. Our model allows retroactive action progression and concurrency (i.e., action overlap) of both discrete and continuous change, including concurrent effects to the same variable. We define precise semantics, including a transition function to compute the result of executing an action (with new temporal properties) in a model and a solution for reasoning about this temporal planning problem. We argue that MPMTL is a natural, flexible representation for reasoning about temporal problems requiring continuous change with metric temporal properties in conditions, effects, constraints, and goals.

Related Work

Allen developed a temporal logic (Allen 1983) that defines a set of thirteen possible *qualitative* relationships among two time intervals (e.g., before, meets, overlap) to represent and reason about actions and events based on their duration (Allen 1983; 1994; Allen and Ferguson 1994). This formalism did not consider *metric* temporal properties (e.g., the goal must be achieved by time t_g for t_l time units) or (non-time) continuous real variables like ours in this paper.

Dechter et al. (1991) extended temporal networks to include continuous variables for time. They describe a framework for checking the consistency of temporal constraints with variable time points, rather than for progressing under discrete and continuous effects as in our formalism. Such checking is polynomial time for a *simple temporal network* but it is NP-hard in general. Coles et al. (2010; 2009) used linear programming to handle continuous linear numeric change in their temporal systems POPF and COLIN.

The timeline abstraction we present in this paper is inspired by temporal constraint approaches commonly used in integrated planning and scheduling systems (e.g., (Chien *et al.* 2000; De Benedictis & Cesta 2015; Rajan, Py, & Barreiro 2013)). Our approach complements those systems by formalizing a precise semantics for timelines while also modeling continuous, dynamic change.

Other researchers have created (McDermott 1998; 2004; Smith, Frank, and Cushing 2008) or extended (Fox and Long 2006; 2003) languages for expressing problems with different temporal features. Fox and Long (2003; 2006) proposed several extensions to PDDL to represent numeric expressions, actions with fixed-length duration, and discrete effects, as well as modeling continuous state changes through the use of autonomous processes and events. Hoffmann and Edelkamp (2005) proposed a further extension to include domain axioms and timed initial literals. Conversely, Smith et al. (2008) developed the *Action Notation Modeling Language (ANML)*, which supports a limited form of HTN methods and represents finite-duration actions with numeric change succinctly. Our work considers similar features of temporal planning but with *metric* temporal properties; we focus on a formalism rather than a language.

Some researchers focused on translating problems defined in PDDL+ to hybrid automata (Bogomolov *et al.* 2015; Fox and Long 2006) rather than using a temporal logic formalism to reason about the problem, which is what we do in this paper.

Many researchers identified fragments of temporal logics that reduce the complexity of such problems (Bresolin *et al.* 2009; Halpern and Shoham 1996; Monica *et al.* 2011; Sciavicco 2012). Similarly, we define a subset of MPMTL-formulae in a disjunctive normal form (DNF) over *n-literals*.

Scenario: Metric Temporal (Interval) Action Conditions/Effects and Goals

Consider the task of maintaining a continuous communications relay between a distant low-power device (i.e., a cell phone) at location L and a base B, as could happen during disaster recovery or in remote areas. There exists no nearby tower (or it is damaged) and low-powered devices require relatively close proximity to a network access point. A loitering UAV over L provides a connection back to B as long as its fuel supply lasts. If the relay time exceeds the fuel

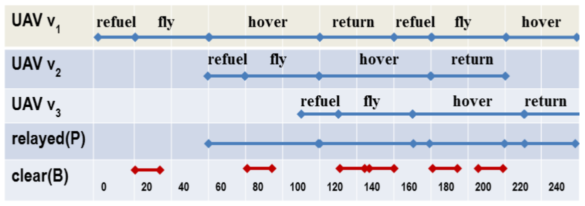

Figure 1: An example of a solution for three UAVs to continuously relay a VIP P, where $t_r = 20$ (refueling time), $t_g = 60$ (required start-by time), and $t_l = 200$ (required duration of relay), assuming each UAV can fly and/or hover for at most 140 time units.

supply, then a schedule for multiple vehicles is required to maintain the relay. Figure 1 displays a plan for a continuous relay maintenance goal using three UAVs. We parameterize a UAV in terms of its refueling time t_r, cruise speed $speed$, fuel level $fuel$, and fuel consumption rate $\delta fuel$. We assume that t_r, $speed$, and $\delta fuel$ are constant for each UAV, leaving only the position of the UAVs and their fuel level as dynamic discrete and continuous variables. The relay must begin by a specficed time t_g after the start of the scenario and the relay is required for a minimum length of time t_l. This relay goal can be expressed as a formula (e.g., $\Diamond_{[0,t_g]}\Box_{[0,t_l]}relayed$) in a metric temporal logic.

Each $relayed$ effect depends on the start time and duration of the individual UAVs performing a $relay$ action, which in turn depends on the distance between B and L. UAVs can be reused after refueling. A condition for a UAV to fly from or to B is that its support team at B is ready for it to take off or land (e.g., for 15 time units). During this time no other UAV can take off from or land to B. This temporal condition, denoted by $clear(B)$, is changed by the *temporal interval* effect of a UAV's fly action to $\neg clear$ for 15 time units. Thus, our planner must permit concurrent actions whose conditions and effects interact. In interval $[120, 180]$ in Figure 1, for example, UAV v_1 flying to B departs before the time UAV v_3 departs from B, but v_3 needs $clear(B)$ and changes it to $\neg clear(B)$ before v_1 needs $clear(B)$.

Preliminaries

We consider only binary variables in $\{0, 1\}$ ($\{\bot, \top\}$) and real variables in $\mathbb{R}$. To ensure practical computation, we assume the linearity of continuous changes (for real variables) and a finite number action effects to a variable. As we will show, these assumptions allow for consistency checking in time polynomial in the size of a variable's timeline.

Time Points and Time Intervals

A *time point* $t \in \mathbb{R}_+$ is a non-negative real value, where $\mathbb{R}_+ = \{x \in \mathbb{R} \mid x \geq 0\}$.

A *time interval (interval)* is a convex subset of $\mathbb{R}_+$ of one of the following forms: $[a, b]$, $[a, b)$, $[a, \infty)$, $(a, b]$, (a, b), or (a, ∞), where $0 \leq a \leq b$ and $a, b \in \mathbb{R}_+$. Intervals can be open, half-open, or closed and bounded or (right) unbounded. For such an interval I, a and b are its left end-point and right end-point, denoted by I^- and I^+, respectively.

An interval I is *singular* if it is of the form $[a,a]$ (i.e., $I^- = I^+ \in I$). Two intervals I and I' are *overlapped* if $I \cap I' \neq \emptyset$, where $I \cap I' = \{t \mid t \in I \wedge t \in I'\}$. I' *continues* I if $I^+ = I'^-$ and either I is right-open ($I^+ \notin I$) and I' is left-closed ($I'^- \in I'$) or I is right-closed and I' is left-open. I and I' are *adjacent* if one continues the other.

The extension of I, denoted by $ext(I)$, is the closed interval $[I^-, I^+]$ if $I^+ < \infty$, or the left closed interval $[I^-, \infty)$, otherwise. $I+t$ denotes the interval $\{t + t' \mid t' \in I\}$, for $t \in \mathbb{R}_+$. Similarly, $I-t$ denotes $\{t' - t \mid t' \in I \wedge t' \geq t\}$. For example, $(2,6] + 3 = (5,9]$ and $[4,10) - 5 = [0,5)$.

The addition of two intervals I and I' is defined as $I + I' = \{t + t' \mid t \in I \wedge t' \in I'\}$. For example, $[2,3] + [1,4] = [3,7]$, while $[2,3) + [1,4] = [2,3) + [1,4) = [3,7)$.

Variables and Literals

A *literal* is a primitive assertion about a variable's value. We start with a binary variable first. Let x be a free variable in $\{0,1\}$. A *binary literal* ℓ of x is a tuple of the form $\langle x, \text{'='}, b\rangle$, where $b \in \{0,1\}$. The negation of ℓ, denoted by $\neg\ell$, is the literal $\langle x, \text{'='}, 1-b\rangle$. We will often omit the quotation marks from the relation operator (e.g., '$=$' is often written as $=$) when the context is clear.

We extend the definition of a literal to variables in $\mathbb{R}$ with the relational operators $O_r = \{=, \neq, <, >, \leq, \geq\}$, which are often used for real numbers. Let $r \in O_r$. The *inverse* of r, denoted by $\bar{r}$, is defined as

r	$=$	$\neq$	$<$	$>$	$\leq$	$\geq$
$\bar{r}$	$\neq$	$=$	$\geq$	$\leq$	$>$	$<$

Let x be a free variable in $\mathbb{R}$. A *continuous literal* ℓ of x is a tuple of the form $\ell = \langle x, r, a\rangle$, where $r \in O_r$ and a is a constant in $\mathbb{R}$ or a pair (b,c) of constants in $\mathbb{R}$, $b \neq 0$. The negation $\neg\ell$ of ℓ is the literal $\neg\ell = \langle x, \bar{r}, a\rangle$.

Literal $\ell = \langle x, r, a\rangle$ (or '$x\ r\ a$') is *constant* if a is a single value (either boolean or real). Intuitively, a constant literal denotes the value of a variable or its relation with a constant value (e.g., $x = 1$ ($x = \top$), $x \geq 10.12$). In the following we specify literals derivable from others (of the same variable).

Let $\ell = \langle x, r, a\rangle$ and $\ell' = \langle x, r', b\rangle$ be two constant literals of the same variable x. We say that ℓ *implies* ℓ', denoted by $\ell \Rightarrow \ell'$, iff $\ell = \ell'$ or $a, b \in \mathbb{R}$ and ℓ implies ℓ' with the usual meaning. That is:

- $x > a$ implies $x > b$, $x \neq b$, and $x \geq b$, where $a \geq b$
- $x \geq a$ implies $x > b$ and $x \neq b$, where $a > b$
- $x \geq a$ implies $x \geq b$, where $a \geq b$
- $x < a$ implies $x < b$, $x \neq b$, and $x \leq b$, where $a \leq b$
- $x \leq a$ implies $x < b$ and $x \neq b$, where $a < b$
- $x \leq a$ implies $x \leq b$, where $a \leq b$
- $x = a$ implies $x \diamond b$, where $a \diamond b$ (for $\diamond \in O_r$)

We note that the "imply" relation is transitive.

We say ℓ and ℓ' *contradict*, denoted by $\ell \not\Leftrightarrow \ell'$, iff $\ell \Rightarrow \neg\ell'$ (or $\ell' \Rightarrow \neg\ell$), e.g., $x > 3 \not\Leftrightarrow x = 2$ as $x > 3 \Rightarrow x \neq 2$.

Interval Literals (i-literals)

The semantics of a non-constant literal (whose third component is a pair (b,c) of real constants) are defined in association with a time interval as follows:

Definition 1. *An* interval literal *or* i-literal *of a variable x is a tuple of the form $\langle \ell, I\rangle$, where $\ell = \langle x, r, a\rangle$ is a literal of x and I is a time interval. The* (instant) literal *of the i-literal $\langle \ell, I\rangle$ at a time point $t \in ext(I)$, denoted by $\langle \ell, I\rangle(t)$, is a constant literal, defined as:*

$$\langle \ell, I\rangle(t) = \begin{cases} \ell & \text{if } \ell \text{ is constant} \\ \langle x, r, b(t-I^-)+c\rangle & \text{if } a \text{ is a pair } (b,c) \text{ in } \mathbb{R} \end{cases}$$

The line segment *of the i-literal $\langle \ell, I\rangle$, denoted by $\Delta(\ell, I)$, is defined by the following set of points:*

$$\Delta(\ell, I) = \begin{cases} \{(a,t) \mid t \in I\} & \text{if } \ell \text{ is constant} \\ \{(b(t-I^-)+c, t) \mid t \in I\} & \text{if } a \text{ is a pair } (b,c) \end{cases}$$

Intuitively, an i-literal $\langle \ell, I\rangle$ denotes a relation r of a variable x with respect to its line segment $\Delta(\ell, I)$ in a plane, in which the vertical axis represents variable x and the horizontal axis represents time. For example, the i-literal $\langle z > (2,-1), [2,7]\rangle$ means that if $z(t)$ is the value of z at time $t \in [2,7]$ then $z(t) > -1 + 2(t-2)$ and the point $(z(t), t)$ must be above its **green** line segment in Figure 3 as shown later.

Two i-literals are adjacent (overlapped) if their time intervals are adjacent (overlapped). Let $\langle \ell, I\rangle$ and $\langle \ell', I'\rangle$ be two i-literals of the same variable. We say that $\langle \ell, I\rangle$ *implies* $\langle \ell', I'\rangle$, denoted by $\langle \ell, I\rangle \Rightarrow \langle \ell', I'\rangle$, iff $\forall t \in I' \mid t \in I \wedge \langle \ell, I\rangle(t) \Rightarrow \langle \ell', I'\rangle(t)$. Finally, $\langle \ell, I\rangle$ and $\langle \ell', I'\rangle$ *contradict* iff $I \cap I' \neq \emptyset \wedge \exists t \in I \cap I' \mid \langle \ell, I\rangle(t) \not\Leftrightarrow \langle \ell', I'\rangle(t)$.

i-literal implication or contradiction Let $\ell = \langle x, r, a\rangle$ and $\ell' = \langle x, r', a'\rangle$ be two literals of variable x. The following proposition shows that checking whether an i-literal implies another can be (sufficiently) performed at the two end-points of its interval (or, if the interval is unbounded, we just need to pick a very large number for its right endpoint).

Proposition 1. *Let $\langle \ell, I\rangle$ and $\langle \ell', I'\rangle$ be two i-literals such that $I' \subseteq I$. ($\forall t \in I' \mid t \in I$.) Let ℓ_l and ℓ_r be the instant literals of i-literal $\langle \ell, I\rangle$ at the left and right endpoints of I' ($\ell_l = \langle \ell, I\rangle(I'^-)$ and $\ell_r = \langle \ell, I\rangle(I'^+)$). Let ℓ'_l and ℓ'_r be the instant literals of i-literal $\langle \ell', I'\rangle$ at the left and right end-point of I'. Then $\langle \ell, I\rangle \Rightarrow \langle \ell', I'\rangle$ iff:*

1. *$r' \neq$ '$\neq$' $\wedge\ \ell_l \Rightarrow \ell'_l \wedge \ell_r \Rightarrow \ell'_r$, or*
2. *$r' \in \{<, >\} \wedge$*
   $(\ell_l \Rightarrow \ell'_l \vee (\ell_l \Rightarrow \langle x = a', I'\rangle(I'^-) \wedge I'^- \notin I')) \wedge$
   $(\ell_r \Rightarrow \ell'_r \vee (\ell_r \Rightarrow \langle x = a', I'\rangle(I'^+) \wedge I'^+ \notin I'))$, or
3. *$r' =$ '$\neq$' $\wedge\ (\langle \ell, I\rangle \Rightarrow \langle x > a', I'\rangle \vee \langle \ell, I\rangle \Rightarrow \langle x < a', I'\rangle)$*

Note that case 3 of Proposition 1 is deduced from case 2.

We can also check whether two i-literals contradict at one time point. Let I and I' be two time intervals.

Proposition 2. *Then two i-literals $\langle \ell, I\rangle$ and $\langle \ell', I'\rangle$ contradict iff $I \cap I' \neq \emptyset$ and:*

- *$\forall t \in I \cap I', \langle \ell, I\rangle(t)$ and $\langle \ell', I'\rangle(t)$ contradict (i.e., $\langle \ell, I \cap I'\rangle \Rightarrow \langle \neg\ell', I \cap I'\rangle$), or*
- *The two line segments $\Delta(\ell, I)$ and $\Delta(\ell', I')$ intersect at a unique point (v,t) ($t \in I \cap I'$) and $\langle \ell, I\rangle(t) \not\Leftrightarrow \langle \ell', I'\rangle(t)$.*

Then it follows directly from Propositions 1 and 2 that,

Proposition 3. *Checking either $\langle \ell, I\rangle \Rightarrow \langle \ell', I'\rangle$ or $\langle \ell, I\rangle \not\Leftrightarrow \langle \ell', I'\rangle$ requires constant time.*

Timelines

Let x be a variable in the domain $\{0,1\}$ or $\mathbb{R}$. A *timeline* μ of x is a set of i-literals of the form $\langle x=a, I\rangle$ such that (1) μ does not contain a pair of i-literals that contradict and (2) $\bigcup_{\langle x=a,I\rangle\in\mu} I = \mathbb{R}_+$. For example, in Figure 1 we assume the fuel capacity of a UAV is 140, $\delta fuel{=}-1$. Then the timelines for $fuel(v_2)$ (f_2) and $relay(P)$ (r_p) are as:

- $\{\langle f_2=0,[0,60]\rangle, \langle f_2=(7,0),(60,80]\rangle,$ $\langle f_2=(-1,140),(80,220]\rangle, \langle f_2=0,(220,\infty)\rangle\}$
- $\{\langle r_p{=}0,[0,60)\rangle, \quad \langle r_p{=}1,[60,120]\rangle, \quad \langle r_p{=}1,[120,180]\rangle,$ $\langle r_p{=}1,[170,230]\rangle, \langle r_p{=}1,[220,260]\rangle, \langle r_p{=}0,(260,\infty]\rangle\}$

Using a timeline, we model a variable's behavior in a sequence of time intervals, where in each interval its value remains constant or linearly varies (if it represents real values). For each time point t, there exists an i-literal $\langle \ell, I\rangle$ in μ such that $t\in I$ and the value of the variable at t can be defined as the third component of $\langle \ell, I\rangle(t)$.

Let x be a variable in $\{0,1\}$ or $\mathbb{R}$ and μ be a timeline of x. Let t be a time point and $\langle \ell, I\rangle$ be an i-literal in μ such that $t\in I$. The *(instant) literal* of timeline μ at time point t, denoted by $\mu(t)$, is defined by the literal $\mu(t)=\langle \ell, I\rangle(t)$.

The (instant) value of x at time point t in μ, denoted by $x(\mu,t)$, is defined as the third component of the literal $\mu(t)$ (i.e., $x(\mu,t)=c$, where $x=c$ is the literal $\mu(t)$).

Two i-literals $\langle \ell, I\rangle$ and $\langle \ell', I'\rangle$ of the same variable *can be merged* iff (1) I and I' are overlapped or adjacent and (2) if $I^-{\le}I'^-$ then $\langle \ell, I{+}I'\rangle{\Rightarrow}\langle \ell', I'\rangle$ (e.g., $\langle \ell, I{+}I'\rangle$ can replace the two i-literals) or vice versa. A timeline's size $|\mu|$ is the number of i-literals in μ. A timeline is *minimal* if it does not contain a pair of overlapped i-literals and it does not contain any timelines that could be merged. Timelines μ and μ' are *equivalent*, denoted by $\mu\equiv\mu'$, if $\forall t\in\mathbb{R}_+ \mid \mu(t){=}\mu'(t)$. In the above examples, the first timeline is minimal, but the second one is not and a minimal timeline equivalent to it is $\{\langle r_p{=}0,[0,60)\rangle, \langle r_p{=}1,[60,260]\rangle, \langle r_p{=}0,(260,\infty]\rangle\}$.

Proposition 4. *Let μ be a minimal timeline and μ' be another timeline equivalent to μ. It holds that $|\mu'|\ge|\mu|$ and if $|\mu'|=|\mu|$ then μ' is also minimal.*

We assume timelines are minimal unless otherwise stated.

We next show the *satisfaction* of an i-literal in a timeline of the same variable. Let x be a variable and μ be a timeline of x. Let ℓ be a literal of x and I be a time interval. The i-literal $\langle \ell, I\rangle$ is said to be *satisfied* or *true* in μ, denoted by $\mu\models\langle \ell, I\rangle$, if $\forall t{\in}I \mid \mu(t){\Rightarrow}\langle \ell, I\rangle(t)$. We also say that $\langle \ell, I\rangle$ is *false* in μ, denoted by $\mu\not\models\langle \ell, I\rangle$, if it is not true in μ.

One can prove the following proposition, which shows how to check satisfaction of an i-literal in a timeline.

Proposition 5. *Let x be a variable in $\{0,1\}$ or $\mathbb{R}$, μ be a timeline of x, ℓ be a literal of x, and I be an interval. Then:*

1. *$\mu\models\langle \ell, I\rangle$ iff $\nexists\langle \ell_i, I_i\rangle\in\mu \mid \langle \ell_i, I_i\rangle\not\Rightarrow\langle \ell, I\rangle$*
2. *Checking whether $\mu{\models}\langle \ell, I\rangle$ requires time linear in $|\mu|$.*

Mixed Propositional Metric Temporal Logic

We are now ready to define a formula in MPMTL. Let ℓ be a literal, I be a time interval, and t be a time point. An *MPMTL-formula* φ is defined by the following grammar:

$$\varphi ::= \top \mid \bot \mid \langle \ell, I\rangle \mid \neg\varphi \mid \varphi\wedge\varphi \mid \varphi\vee\varphi \mid \Box_I\varphi \mid \Diamond_I\varphi$$

By definition, MPMTL formulae are constructed based on i-literals and the constants $\top$ ($true$) and $\bot$ ($false$), as primitive literals. As in other metric temporal logics, the (metric) modalities *always* $\Box_I$ and *eventually* $\Diamond_I$, in formulas $\Box_I\varphi$ and $\Diamond_I\varphi$, mean φ will be true after t time units for *all* $t\in I$ and for *some* $t\in I$, respectively. For the precise semantics of these operators, we need the following:

Definition 2. *Let μ be a timeline and t be a time point. The* left-shift *of μ by t, denoted by $\mu-t$, is a timeline defined as:*

$$\mu-t=\{\langle \ell, I-t\rangle \mid \langle \ell, I\rangle\in\mu\wedge I-t\neq\emptyset\}$$

The semantics of the left shift operation for timelines is shown in the following proposition:

Proposition 6. *Let ℓ be a literal, μ be a timeline. Let t and t' be time points such that $t'{\ge}t$. Then, $(\mu{-}t)(t'{-}t)=\mu(t')$*

We are now ready to define the semantics and a model of an MPMTL-formula as follows:

Definition 3. *Let $\vec{x}=\langle x_1,\dots,x_n\rangle$ be a vector of variables in $\{0,1\}$ or in $\mathbb{R}$. A vector $M=\langle\mu_1,\dots,\mu_n\rangle$ is a* timeline vector *of $\vec{x}$ if each μ_i is a timeline of x_i, for $i=1,\dots,n$. Let $M-t=\langle\mu_1-t,\dots,\mu_n-t\rangle$ for $t\in\mathbb{R}_+$. Let ℓ_i be a literal of variable x_i and φ be an MPMTL-formula built over literals of variables in $\vec{x}$. We say that φ is* true *in M, or M is a* model *of φ, denoted by $M\models\varphi$, iff:*

- *$\varphi=\top$*
- *$\varphi=\langle \ell_i, I\rangle$ and $\mu_i\models\langle \ell_i, I\rangle$ (recall that ℓ_i and μ_i are of the same variable x_i)*
- *$\varphi=\neg\psi$ and $M\not\models\psi$*
- *$\varphi=\psi\wedge\phi$ and $M\models\psi\wedge M\models\phi$*
- *$\varphi=\psi\vee\phi$ and $M\models\psi\vee M\models\phi$*
- *$\varphi=\Box_I\psi$ and $\forall t\in I \mid M-t\models\psi$*
- *$\varphi=\Diamond_I\psi$ and $\exists t\in I \mid M-t\models\psi$*

A formula φ is *satisfiable* iff it has a model. We have:

Proposition 7. *Let $\vec{x}$ be a vector of variables in $\{0,1\}$ or $\mathbb{R}$, φ be an MPMTL-formula built over literals of variables in $\vec{x}$, and M be a timeline vector of $\vec{x}$. Then:*

1. *Checking whether φ is satisfiable is undecidable*
2. *Checking whether M is a model of φ is undecidable*

These are undecidable due to the dense time and nested modalities allowed in MPMTL formulas. Therefore, we need to find a fragment of MPMTL formulas that is computationally practical.

DNF, Γ, and Γ_n Fragments

We identify the three fragments of MPMTL and a method for transforming a general MPMTL-formula to one in DNF class and analyze the complexity of basic operations over formulae in DNF, Γ, and Γ_n.

Let φ and ψ be two MPMTL-formulae, and $M(\varphi)$ be the set of models of φ. We say that φ *implies* ψ ($\varphi\Rightarrow\psi$) iff every model of φ is a model of ψ (i.e., $M(\varphi)\subseteq M(\psi)$) and φ and ψ are *equivalent* ($\varphi\equiv\psi$), iff $M(\varphi)=M(\psi)$. As in classical propositional logic, in this formalism the connectives $\wedge$ and $\vee$ are distributive and commutative. By definition, we can also deduce the following:

Proposition 8. *Let ℓ be a literal, φ and ψ be two MPMTL-formulae, and I and I' be two time intervals. Then:*

1. *$\Box_I\langle\ell, I'\rangle \equiv \bot$ (unsatisfiable), iff ℓ is of the form $\langle x, =, (b,c)\rangle$ and both I and I' are non-singular, and $\Box_I\langle\ell, I'\rangle \equiv \langle\ell, I{+}I'\rangle$, otherwise*
2. *$\neg\Box_I\varphi \equiv \Diamond_I\neg\varphi$ and $\neg\Diamond_I\varphi \equiv \Box_I\neg\varphi$*
3. *$\Box_I\Box_{I'}\varphi \equiv \Box_{I+I'}\varphi$ and $\Diamond_I\Diamond_{I'}\varphi \equiv \Diamond_{I+I'}\varphi$*
4. *$\Box_I(\varphi\wedge\psi) \equiv \Box_I\varphi\wedge\Box_I\psi$ and $\Diamond_I(\varphi\vee\psi) \equiv \Diamond_I\varphi\vee\Diamond_I\psi$*
5. *$\Box_I\varphi\vee\Box_I\psi \Rightarrow \Box_I(\varphi\vee\psi)$ and $\Diamond_I(\varphi\wedge\psi) \Rightarrow \Diamond_I\varphi\wedge\Diamond_I\psi$*
6. *$\Box_I(\varphi \vee \psi) \equiv \Box_I\varphi \vee \Box_I\psi$, if φ and ψ do not share a variable*

We call an *n-order literal or n-literal* a formula φ of the form $\bigcirc_1 \ldots \bigcirc_{n-1}\ \varphi$, where φ is an i-literal or its negation and $\bigcirc_i$ denotes $\Box_I$, $\neg\Box_I$, $\Diamond_I$, or $\neg\Diamond_I$. In this case, we say that φ has order n, the number of modalities it contains (including an implicit one in the i-literal). Due to laws 1-3 in Proposition 8, such a literal can be reduced to an equivalent lower order literal (of fewer modalities) by flipping $\neg\Box_I$ to $\Diamond_I$ and $\neg\Diamond_I$ to $\Box_I$ and combining consecutive modalities of the same type (law 3). For example, $\varphi = \Box_I\neg\Diamond_{I'}\neg\langle\ell, I"\rangle \equiv \Box_I\Box_{I'}\langle\ell, I"\rangle$ (i.e., either $\varphi \equiv \bot$ or $\varphi \equiv \langle\ell, I_a\rangle$ (law 1), where $I_a = I{+}I'{+}I"$). Also observe that a n-literal of the form $\ldots\Box_{I_{n-2}}\Diamond_{I_{n-1}}\langle\ell, I'\rangle$ or $\ldots\Diamond_{I_{n-2}}\Box_{I_{n-1}}\neg\langle\ell, I'\rangle$, in which two consecutive modalities are of different types and there is no negation to the left of a modality, cannot be reduced to a strictly lower order literal. We call these forms a *normal form* and a literal of one of these forms a *normalized* literal.

Proposition 9. *Any n-literal can be reduced to an equivalent* unique normalized *m-literal in $O(n)$ time such that $m \le n$.*

An MPMTL-formula φ is said to be a *disjunctive normal form (DNF)* formula iff φ is a disjunction of conjunctions of normalized literals. We define the order of an MPMTL formula to be the maximum number of nested modalities in it. Thus, the order of a DNF-formula is the highest order of a literal in it. Let Γ be the set of MPMTL-formulae that can be reduced to a DNF formula φ of order ≤ 2 such that if a conjunction of φ contains two or more order literals of the same variable then none of them has order greater than 1.

Lemma 1. *Let α be a 2-literal and μ be a timeline of the variable in α. Let φ be a formula of n literals in Γ. Then:*

1. *Checking whether $\mu \models \alpha$ is $O(|\mu|^2)$ time*
2. *Checking whether φ is satisfiable is $O(n^2)$ time*

Proof. (Sketch)

1. Due to Proposition 9, it is sufficient to prove for normalized 2-literals, i.e., α is of the form $\Diamond_I\langle\ell, I'\rangle$ or $\Box_I\neg\langle\ell, I'\rangle$. We start with the case where $\alpha = \Diamond_I\langle\ell, I'\rangle$, $\ell = x \ge (b,c)$, and $b > 0$. In this case, $\mu \models \alpha$ iff $\exists t \in I$ such that $\mu \models \langle\ell, I'+t\rangle$, i.e., the line segment $\Delta(\ell, I'+t)$ is bellow the part of any segment in μ in interval $I'+t$. For some $t \in I$, this can be verified at the endpoints of $I'+t$ and of other intervals in μ in $I'+t$ (Proposition 1, case 1) in $O(|\mu|)$ time. Now we construct a set S_t of such relative times t. Let S_0 be the set of endpoints of I and of other intervals in μ in I. Note that the minimum (resp. maximum) vertical value of a point in $\Delta(\ell, I'+t)$ is at its left (resp. right) endpoint, which is c (resp. $b(I'^{+} - I'^{-}) + c$). Let S_1 be the set of times at the intersections of line $x = c$ and line segments in μ such that $\forall t' \in S_1 \mid t' \in I$. Let S_2 be the set of times at the intersections of line $x = b(I'^{+} - I'^{-}) + c$ and line segments in μ such that $\forall t' \in S_2 \mid t' - (I'^{+} - I'^{-}) \in I$. Let $S_t = S_0 + S_1 + S_2$, clearly $|S_t| = O(|\mu|)$ and one can also verify that it is sufficient to check whether $\mu \models \alpha$ with relative times in S_t in $O(|\mu|^2)$ time. Other cases can be proved similarly.
2. Let φ_i be a set of all order literals of the same variable in φ. Then φ is satisfiable iff each φ_i is satisfiable. If φ_i contains a literal of order greater than 1, then it is the only order literal in φ_i so it is always satisfiable. Otherwise, φ_i contains a set of at most n i-literals. For each pair of i-literals of the forms $\langle x \ge a, I\rangle$ and $\langle x \le a', I'\rangle$ such that their line segments have a common segment $\langle x = a", I \cap I'\rangle$, we add this segment to φ_i (for checking contradiction of i-literal of relation $\neq$). One can verify that φ_i is satisfiable iff it does not contain a pair of i-literals that contradict. This can be checked in $O(n^2)$. ☐

Theorem 1. *Let φ be a DNF-formula of n order literals. Let M be a timeline vector of variables in φ and m be the maximum size of a timeline in M. Then:*

1. *Checking the satisfiability of φ is $O(n^2)$, if φ is in Γ*
2. *Checking whether M is a model of φ is $O(nm)$, if φ has order 2 or less*

Proof. This follows directly from Lemma 1 ☐

The low complexity in Theorem 1 is due to the low order and the special structure of the DNF formula, which allows checking each order literal in the formula individually. (The same problems for a DNF formula in traditional propositional logic are linear.)

The following theorem identifies the set of MPMTL-formulae that can be reduced to a DNF-formula and the upper bound of the complexity of the transformation and the size of the resulting formula.

Proposition 10. *Let φ be an MPMTL-formula such that φ does not contain a sub-formula of the form $\Box_I\phi \vee \Box_I\psi$, where ϕ and ψ share a variable, or $\Diamond_I(\phi \wedge \psi)$. Let n be the number of i-literals in φ, and k be the order of φ. Then:*

1. *φ can be reduced to a DNF-formula φ' of order $q \le k$ by using distributivity of the connectives $\wedge$ and $\vee$ and laws 1-4 and 6 in Proposition 8.*
2. *The transformation of φ into the DNF-formula φ' can be performed in $O(k \times 3^{n/3})$ time.*
3. *φ' contains at most $(3^{n/3})$ order-literals.*

Proof. (Sketch) We prove items 2 & 3. The transformation can be done as follows. First, we move all modalities and negation outside a parenthesis to be in front of the i-literals and converts them to normalized form, e.g., $\Box_{I_1}\neg(\Diamond_{I_2}p \vee \Box_{I_3}(\Diamond_{I_4}q\wedge k)) \equiv \Box_{I_1+I_2}\neg p\wedge(\Box_{I_1}\Diamond_{I_3}\Box_{I_4}\neg q\vee\Box_{I_1}\Diamond_{I_3}\neg k)$.

This takes $O(kn)$ time and produces a special negation normal form formula φ" of n normalized literals. The conversion of this formula to DNF form is the same to that for a negation normal form formula in classical logic. One can prove by induction on n the resulting DNF formula has most $(3^{n/3})$ literals. For example, if $n = 9$ then the worst case is that φ" has the form $(x_1 \vee x_2 \vee x_3) \wedge (x_4 \vee x_5 \vee x_6) \wedge (x_7 \vee x_8 \vee x_9)$ and φ' contains $27 = 3^{9/3}$ literals. □

Let Γ_n be the set of MPMTL-formulae that can be reduced to a DNF formula of order n or less. Observe that $\Gamma_n \subset \Gamma_m$ iff $n < m$. We then have:

Theorem 2. *Let φ be an MPMTL-formula and n be the number of i-literals in φ. Let M be a timeline vector of variables in φ and m be the maximum size of a timeline in M. Then:*

1. *Checking the satisfiability of φ is $O(3^{2n/3})$, if φ is in Γ*
2. *Checking whether M is a model of φ is $O(m3^{n/3})$, if φ is in Γ_2*

Proof. This follows from Theorem 1 and Prop. 10 □

One can see that the fragments Γ and Γ_2 contains many complex MPMTL-formulae of order greater than 2.

Temporal planning with MPMTL

We present a propositional model for temporal planning based on MPMTL. Most of the new features introduced in this formulation will be contrasted with the most common temporal planning languages, including PDDL2.1 (Fox and Long 2003), PDDL+ (Fox and Long 2006), and ANML (Smith, Frank, and Cushing 2008).

We consider only grounded problems, in which propositions and functions (of continuous values, defined in PDDL languages) are represented by binary and real variables, respectively. For brevity, we write the literals $x = 1$ and $x = 0$ of a binary variable x as x and $\neg x$, respectively.

Action, Constraint, and World Model

We begin with an example of an action (schema) for a UAV v to fly from location l_1 to location l_2

Action: $fly(v, l_1, l_2)$
Duration: $d = distance(l_1, l_2)/speed(v)$
Condition: $\Box_{[t_s,t_s]}(\,\langle at(v, l_1), [0, d]\rangle\wedge$
$\langle\neg flying(v), [0, d]\rangle \wedge \langle clear(l_1), [0, 15]\rangle\wedge$
$\langle clear(l_2), [d{-}15, d]\rangle)$
Constraint:
$\Box_{[t_s,t_s]}(\langle flyable(v), [0, d]\rangle \wedge \langle fuel(v){>}0, [0, d]\rangle)$
Effect: $\{\langle\neg at(v, l_1), [0, 0]\rangle^+, \langle at(v, l_2), [d, d]\rangle^+,$
$\langle flying(v), [0, d]\rangle, \langle\neg clear(l_1), [0, 15]\rangle, \langle\neg clear(l_2),$
$[d{-}15, d]\rangle), \langle fuel(v) \mathrel{+}= (\delta fuel(v), 0), [0, d]\rangle)\}$

In this example, t_s and $\delta fuel(v)$ denote the action's start time and UAV v's fuel consumption rate, respectively. The condition and constraint are formulae in Γ_2, while the effect is a set of extended i-literals. The end-points of an interval in these components may depend on the duration d and those of the leftmost interval depend on the action start time (t_s). An interval in the effect is relative to the start time of the action (e.g., the actual interval in the i-literal $\langle flying(v), [0, d]\rangle$ would be $[0, d]{+}t_s = [t_s, t_s{+}d]$). Here, the duration d is fixed because the distance between two locations l_1 and l_2 and v's speed are given. However, an action may have a variable duration, often restricted by some constraint. The condition $\langle clear(l_1), [0, 15]\rangle$ specifies that location l_1 must be clear over interval $[t_s, t_s + 15]$ for v to take off from l_1. The effect $\langle fuel(v) \mathrel{+}= (\delta fuel(v), 0), (0, d]\rangle)$ specifies that the UAV's fuel level will decrease linearly by its consumption rate $\delta fuel(v)$ during its flight.

Unlike most approaches that adopt state models of the world, ours represents the world by a (1) timeline vector (*world model*) and (2) a set of constraints on the variables defined in the problem. We call such a pair a *constrained world model (c-model).* The execution of an action will change the world model instead of a state. This allows the world to be represented as a continuous process, relevant to temporal planning with continuous change, rather than as a (sequence of) discrete *snapshot(s)*, as in other approaches.

For a precise model of temporal properties and the semantics of actions we distinguish two types of action effects: one changes the value of a variable as a function of its current value (e.g., $\langle x = \neg x, I\rangle$ or $\langle x \mathrel{+}= a, I\rangle$), while the other assigns an absolute value to the variable (e.g., $\langle x = (2, 5), I\rangle$). We call these *relative* and *absolute* effects, and the time point at which a variable is set with an absolute value a *fixed point*. There are two possible approaches to absolute effects: (1) No other effect to the same variable that changes the value set by an absolute effect at the fixed point is allowed, and (2) such an effect is not necessarily forbidden but the (absolute) value at the fixed point will remain the same. We choose the latter approach since it is more flexible, though if we want to forbid such an effect, we still can set a constraint in its action.

A variable is called *inertial* if its value remains unchanged after an action effect, until another effect changes it.

Assumptions and Rules

To allow retroactive actions and effects and their overlaps, including concurrent effects to the same variable, we adopt the following assumptions and rules:

1. A variable can change only by actions, and the effect of an action never starts before the instant the action starts (t_s). An effect of a variable x may change its value during and possibly after its interval, but it does not affect its value before the interval.

2. An action is executable in a world model only if (1) its condition is true in the model and (2) all constraints, including that of the action, hold in the resulting model after the action's execution.

3. We forbid two effects of a **binary** variable x that start at the same time, unless they are both absolute and assign the same value to x.

4. Two absolute effects of a variable x that simultaneously assign different values to it are not allowed.

5. Let x be a **binary** variable. An effect to x overrides those that start before it. If x is inertial and its value changes by the effect at a time point then it will negate the outcome of all relative effects that start after it and before any absolute

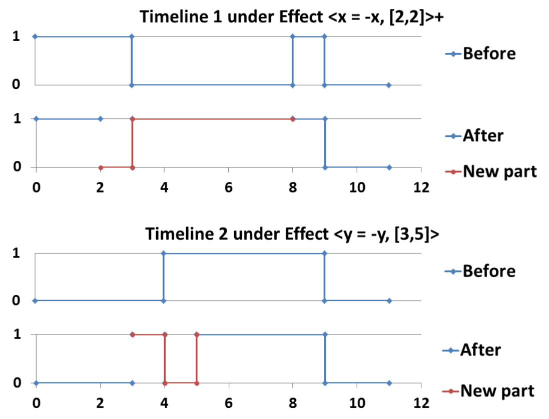

Figure 2: Update timelines μ_1 & μ_2 of binary variables w.r.t. the first two effects (Example 1, presented later) following rule 5. The extendable effect $\langle x = \neg x, [2, 2]\rangle^+$ flips (the value of) x at time 2 of μ_1 and extends this value until the next effect starts at time 3. Then it flips x from time 3 and before the fixed point 8 of absolute effect $\langle x, [8, 9)\rangle^*$ (i-literal $\langle \neg x, [3, 8)\rangle$). This results in the new part in red color ($\langle \neg x, [2, 3)\rangle$ & $\langle x, [3, 8)\rangle$). The value of x before 2 and after the fixed point 8 remains unchanged. The 2nd (relative) effect flips y (in μ_2) only in its interval $[3, 5]$ as it is non-extendable.

effect after it. The value of x set by an effect will extend beyond the interval until it is set by another thereafter, if x is inertial, and it is limited to the effect interval, otherwise. We denote by the superscript '+' an *extendable* effect to a binary inertial variable. In our example of the fly action, $\langle \neg at(v, l_1), [0, 0]\rangle^+$ is an extendable effect. Figure 2 shows examples of effects to binary variables.

6. Any real variable is inertial. Effects to a real variable x are limited to the following forms: (1) assign a constant or a line segment to x for a time interval (e.g., $\langle x = 5, [0, d]\rangle$, $\langle x = (2, 5), [0, d]\rangle$), (2) add a (possibly negative) constant to x at a time (e.g., $\langle x \mathrel{+}= 2.3, [t, t]\rangle$), (3) add a *linear* continuous change for an interval (e.g., $\langle x \mathrel{+}= (b, 0), [0, d]\rangle$), and (4) a combination of the last two forms (e.g., $\langle x \mathrel{+}= (b, \mathbf{c}), [0, d]\rangle$). Note that the first form is absolute and the rest are relative.

7. An absolute effect to a **real** variable x overrides others (i.e., no effect can change the value of x set by the absolute effect in its interval). The linear continuous changes of relative effects are combined in their intersection(s). For example, during the time a UAV v is flying, another UAV relays fuel to v for the interval I with the rate $rfuel$. The fuel level of v will change with the combined rate $rfuel{+}\delta fuel(v)$ over I. If an effect of x over an interval I increases the value of x by $\delta \in \mathbb{R}$ at the right end point I^+, then its value will be increased by δ over the interval (I^+, ∞), if there is no absolute effect to x after I^+, or over the interval (I^+, t_f), if t_f is the smallest fixed point of x after I^+. Figure 3 shows how an effect to a real variable change its timeline.

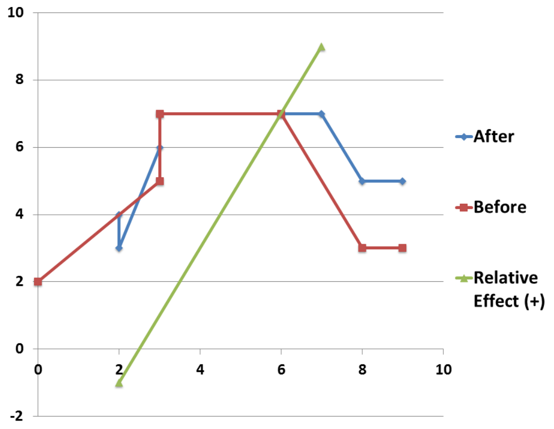

Figure 3: Update timeline μ_3 of real variable z w.r.t. effect $\langle z{+}{=}(2, -1), [2, 7)\rangle$, denoted by the green (line) segment, during and after its interval (Example 1). The red segments represent timeline μ_3 and the blue ones denote the new i-literals in the resulting timeline. The value of z over $[0, 2)$ (before the effect) remains unchanged. The value of z at time 2 reduces by 1 (adds -1) from 4 to 3 and then continuously changes with the combined rate $1 + 2 = 3$ to 6 at time 3. It then remains unchanged over the fixed points $[3, 6)$, despite the effect in this interval. Over the rest ($[6, 7)$) of the interval of the effect, z changes with the combined rate $-2 + 2 = 0$ from value 7 (i.e., it remains unchanged over $[6, 7)$). Since z increases by 2 by the effect at its right endpoint 7, it increases by 2 for all time points after 7 (and before any fixed point after 7). This results in two i-literals $\langle z{=}(-2, 7), [7, 8)\rangle$ and $\langle z{=}5, [8, \infty)\rangle$ **above** their predecessors $\langle z{=}(-2, 5), [7, 8)\rangle$ and $\langle z{=}3, [8, \infty)\rangle$.

MPMTL Temporal Planning Problem

We use Γ_2 or DNF formulae of order 2 to represent conditions and constraints. For checking satisfiability, we use Γ to describe goals. An MPMTL temporal planning problem P is defined as a tuple of the form $\langle \vec{x}, A, M_I, C_I, G\rangle$, where $\vec{x}$ is a vector of binary and real variables, A is a set of actions, M_I is a timeline vector of $\vec{x}$, C_I is a (possibly empty) set of formulae in Γ_2 over $\vec{x}$ denoting the initial constraint, and G is a formula in Γ_2 over $\vec{x}$ denoting the goal.

An action (schema) $a \in A$ is composed of a *duration* d, a *condition* $cond(a)$, a *constraint* $C(a)$, and an *effect* eff(a). The duration d can be either a given constant (fixed, possibly parameterized by some constant properties (e.g., $d = distance(l_1, l_2)/speed(v)$)) or a variable in $\mathbb{R}_+$. A variable duration can be restricted by the (temporal) condition $cond(a)$ and/or constraint $C(a)$. There is no explicit distinction between instantaneous and durative actions, and the duration $d = 0$ does not necessarily mean that action a is instantaneous, as some of its (temporal) condition and/or effect can be after the time the action starts, even with the duration zero ($d = 0$). This description is reasonable for actions like $pull\_trigger$ and $bomb$. The condition $cond(a)$ and the constraint $C(a)$ of action a is an *extension* of an MPMTL formula in Γ_2 of one or more variables in $\vec{x}$ as follows: (1) the end-points of intervals in $cond(a)$ and $C(a)$ can be an expression of duration d and (2) the endpoints of the leftmost intervals in $cond(a)$ and $C(a)$ can be an expression of the start time t_s. The effect eff(a) is a set of extended i-literals of the form $\langle l, I\rangle$, possibly with a superscript '+', where I^- and I^+ can be an expression of the duration d, l can be $x = b$, $x = \neg x$ (only if x is binary), or $x+ = b$ (only if x is real), and b is either a constant in $\{0, 1\}$ (if x is binary) or in $\mathbb{R}$ or a pair of real constants (c, d) (if x is real).

Action Instance and Model Progression

Let a be an action. If the start time and the duration of a are assigned with constants t_s and d, respectively, then the end-points of intervals in the condition $cond(a)$, constraint $C(a)$, and effect eff(a) become constant and, hence, $cond(a)$ and $C(a)$ become formulae in Γ_2. We call such an action, whose start time and duration are assigned with a constant, an *action instance* (or *action*, if there is no need to make such a distinction). Let M be a world model and C be a constraint set. Action (instance) a is *executable* in the c-model (M, C) iff a satisfies all assumptions/rules 2-4.

We need to define a progression function (i.e., the result of executing a in (M, C)). Intuitively, it is a new c-model (M', C'), where C' is the new constraint set $C' = C \cup C(a)$, if a is executable in (M, C). We must also define an *update* function that, given an action a and a model M, returns a new model $M' = update(a, M)$. This can be defined using another function that returns a new timeline $update(e, \mu)$ given an effect e (in eff(a)) and a timeline μ of the same variable, as shown in Algorithm 1. This function can be defined based on rule 5 (for binary variables) and rule 7 (for real variables). We omit a formal definition of this function, since it is straightforward, yet tedious and lengthy. Instead, one can see how this update function works in Example 1.

Algorithm 1 Computing $update(a, M)$

```
1: **Input**: Action instance $a$, timeline vector $M$
2: **Output**: Timeline vector $update(a, M)$
3: Let $X = M$
4: **for** each effect $e$ in eff$(a)$ **do**
5:     Let $\mu$ be the timeline in $X$ of the same variable in $e$
6:     Let $\mu = update(e, \mu)$ {replace $\mu$ with $update(e, \mu)$
       in $X$}
7: **end for**
8: **return** $X$
```

Example 1. *Let M be a model composed of three timelines:*

- $\mu_1 = \{\langle x, [0, 3)\rangle, \langle \neg x, [3, 8)\rangle, \langle x, [8, 9)\rangle^*, \langle \neg x, [9, \infty)\rangle\}$
- $\mu_2 = \{\langle \neg y, [0, 4)\rangle, \langle y, [4, 9)\rangle, \langle \neg y, [9, \infty)\rangle\}$
- $\mu_3 = \{\langle z{=}(1, 2), [0, 3)\rangle, \langle z{=}7, [3, 6)\rangle^*,$ $\langle z{=}(-2, 7), [6, 8)\rangle, \langle z{=}3, [8, \infty)\rangle\}$

where the superscript '' denotes an absolute effect. Let a be an action, where* eff$(a)$$=\{\langle x{=}\neg x, [2, 2]\rangle^+$, $\langle y{=}\neg y, [3, 5]\rangle$, $\langle z{+}{=}(2, -1), [2, 7)\rangle\}$ *(after adding the start time of a to its intervals). The successor model $update(a, M)$ is obtained by updating each effect in* eff(a) *to each timeline of the same variable in M in that order. The resulting model $update(a, M)$ is composed of the following timelines:*

- $\{\langle x, [0, 2)\rangle, \langle \neg x, [2, 3)\rangle, \langle x, [3, 8)\rangle, \langle x, [8, 9)\rangle^*, \langle \neg x, [9, \infty)\rangle\}$
- $\{\langle \neg y, [0, 3)\rangle, \langle y, [3, 4)\rangle, \langle \neg y, [4, 5]\rangle, \langle y, (5, 9)\rangle, \langle \neg y, [9, \infty)\rangle\}$
- $\{\langle z{=}(1, 2), [0, 2)\rangle, \langle z{=}(3, 3), [2, 3)\rangle, \langle z{=}7, [3, 6)\rangle^*,$ $\langle z{=}7, [6, 7)\rangle, \langle z{=}(-2, 7), [7, 8)\rangle \langle z{=}5, [8, \infty)\rangle\}$

Figures 2 & 3 illustrate and explain these update operations.

Due to the above assumptions/rules, one can prove that the following proposition holds.

Proposition 11. *Let a be an action and M be a model. Then $update(a, M)$ does not depend on the order in which the effects in* eff(a) *are applied.*

A progression function that maps pairs of actions and models to models is defined as:

Definition 4. *Let $\vec{x}$ be a vector of variables. Let a be an action (instance), M be a timeline vector, and C be a set of constraints over $\vec{x}$. The result of action a's execution in model (M, C) , denoted by $\Phi(a, M, C)$, is defined as:*

- $(update(a, M), C \cup C(a))$, *if a is executable in (M, C)*
- $\bot$, *otherwise*

Solution to Temporal Planning

For reasoning about the effects of plans, we extend Φ to define $\widehat{\Phi}$, a transition function that maps action sequences and models to models.

Definition 5. *Let $\alpha_i = [a_1, \ldots, a_i]$ be a sequence of i actions. The result of applying a sequence of actions α_n in a model (M, C), denoted by $\widehat{\Phi}(\alpha_n, M, C)$, is defined as*

- *If $n = 0$ then $\widehat{\Phi}([\,], M, C) = (M, C)$;*
- *If $n > 0$ then*
  - *if $\widehat{\Phi}(\alpha_{n-1}, M, C) = \bot$ or a_n is not executable in $\widehat{\Phi}(\alpha_{n-1}, M, C)$, then $\widehat{\Phi}(\alpha_n, M, C) = \bot$*
  - *if $\widehat{\Phi}(\alpha_{n-1}, M, C) \neq \bot$ and a_n is executable in $\widehat{\Phi}(\alpha_{n-1}, M, C)$, then $\widehat{\Phi}(\alpha_n, M, C) = \Phi(a_n, M', C')$, where $(M', C') = \widehat{\Phi}(\alpha_{n-1}, M, C)$*

The order in which actions are progressed in Definition 5 can differ from the order in which they are actually executed (i.e., that of the start time of actions). For example, consider an action a that makes a location l_2 clear as a condition for a $fly(v, l_1, l_2)$ action (for v to land at l_2). In this case, it should be progressed before $fly(v, l_1, l_2)$, but the latter can be executed before the former. We say that an MPMTL formula φ is true in a model (M, C) if $M \models \varphi$.

We now can define a solution to a planning problem as:

Definition 6. *Let $P = \langle \vec{x}, A, M_I, C_I, G\rangle$ be a planning problem. A solution to P is a sequence of action instances in A such that G is true in $\widehat{\Phi}(\alpha_n, M_I, C_I)$.*

Discussion and Limitation

If action conditions and constraints are restricted to sets of extended n-literals, $n \leq 2$, then the progression can be computed in polynomial time. Otherwise, they can be converted one time into DNF-formulae of order 2 or less and the computation is polynomial in the size of these formulae, whose upper bound is exponential in the size of the original formulae (Proposition 10). However, these formulae (for action condition or constraint) are usually small. In the example of the fly action, the condition and constraint are actually a DNF-formula of order 1. On the other hand, the goal in the scenario is actually the 2-literal $\Diamond_{[0,t_g]}\langle relayed, [0, t_l]\rangle$.

Our formalism assumes known time points, which requires known action start time and duration for progression. This may require time discretization for search instead of using techniques for variable time points such as a temporal network (e.g., (Dechter *et al.* 1991)) or linear programming (e.g., (Coles *et al.* 2010)).

Summary and Future Work

We developed a new temporal logic MPMTL as a formalism for temporal planning. We identified fragments of MPMTL, DNF formulae, Γ, and Γ_2 and the complexity of satisfiability and model checking of a formula in these fragments. We presented a new model for temporal planning with complex temporal properties based on the temporal logic we developed. We proposed a new approach to world modeling, relevant to temporal planning with continuous change. Finally, we defined a transition function and solution to the temporal problem for reasoning about this planning problem.

In future work, we will propose a search algorithm and develop a planning system for the temporal planning problem we defined, using the temporal logic we developed here. We will also focus on temporal goals that may require infinite solutions (e.g., maintenance and cyclic goals). It would be useful to extend PDDL for the planning model we presented. Another interesting research direction is to relax our assumptions (e.g., linearity and known time points) and incorporate other techniques such as temporal networks or linear programming in our work.

Acknowledgements

Thanks to OSD ASD (R&E) for sponsoring this research.

References

Allen, J. F., and Ferguson, G. 1994. Actions and Events in Interval Temporal Logic. Technical Report TR521, Computer Science Department, University of Rochester

Allen, J. F. 1984. Towards a general theory of action and time. *Artificial Intelligence*, 23, pp. 123-154.

Allen, J. F. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832-843.

Alur, R. and Henzinger, T. 1993. Real-Time Logics: Complexity and Expressiveness. *Information and Computation*, 104:35-77.

Bacchus, F., and Kabanza, F. 2000. Using temporal logic to express search control knowledge for planning. *Artificial Intelligence*, 116(1-2), 123-191.

Bacchus, F., andKabanza, F. Planning for Temporally Extended Goals. *Annals of Math & Artif. Intel.*, 22, 5-27, 1998.

Barrett, C. W., Sebastiani, R., Seshia, S. A., and Tinelli, C. 2009. Satisfiability modulo theories. IOS Press. 825-885, 2009.

Bauer, A., and Haslum, P. 2010. LTL Goal Specifications Revisited. In *Proc. ECAI*, 2010.

Benton, J., Coles, A., and Coles, Andrew. 2012. Temporal Planning with Preferences and Time-Dependent Continuous Costs. In *ICAPS*, 2012.

Bogomolov, S., Magazzeni, D., Minopoli, S., and Wehrle, M. 2015. PDDL+ Planning with Hybrid Automata: Foundations of Translating Must Behavior. In *ICAPS*, 2015.

Bresolin, D., Goranko, V., Montanari, A., and Sciavicco, G. 2009. Propositional interval neighborhood logics: expressiveness, decidability, and undecidable extensions. *Annals of Pure and Applied Logic*, 161(3):289-304, 2009.

Chien, S., Knight, R., Stechert, A., Sherwood, R., and Rabideau, G. 2000. Using Iterative Repair to Improve the Responsiveness of Planning and Scheduling. In *Proceedings of the 5th International Conference on Artificial Intelligence Planning and Scheduling (AIPS)*, Breckenridge, CO, 2000.

Cimatti, A, . Micheli, A., and Rover, M. 2015. Strong Temporal Planning with Uncontrollable Durations: A State-Space Approach. In *AAAI*, 2015.

Cimatti, A, . Micheli, A., and Rover, M. 2014. Solving strong controllability of temporal problems with uncertainty using SMT. In *AAAI*, 2014.

Coles, Amanda, Coles, Andrew, Fox, M., and Long, D. 2010. Forward-Chaining Partial-Order Planning. In *ICAPS*, 2010.

Coles, Amanda, Coles, Andrew, Fox, M., and Long, D. 2009. Temporal Planning in Domains with Linear Processes. In *IJCAI*, 2009.

Coles, Andrew, Fox, M., Long, D., Smith, A. J. 2008. Planning with Problems Requiring Temporal Coordination. In *AAAI*, 2008.

De Benedictis, R., and Cesta, A. 2015. New Heuristics for Timeline-based Planning. In *Proceedings of the 6th Italian Workshop on Planning and Scheduling*, (pp 33-48), 2015.

Della Penna, G., Intrigila, B., Magazzeni, D,. and Mercorio, F. 2009. UPMurphi: a Tool for Universal Planning on PDDL+ Problems. In *ICAPS*, 2009.

Dechter, R., Meiri, I., and Pearl, J. 1991. Temporal constraint networks. *Artificial Intelligence*, 49, 1991.

Do, M., and Kambhampati, S. 2003. Sapa: A Multi-objective Metric Temporal Planner. *JAIR*, 2003.

Dvorak, F., Bit-Monnot, A., Ingrand, F., and Ghallab, M. 2014. A flexible ANML actor and planner in robotics. In *ICAPS Planning and Robotics Workshop*, 2014.

Fox, M., and Long, D. 2006. Modelling Mixed Discrete-Continuous Domains for Planning. *JAIR*, 27 235-297, 2006.

Fox, M., and Long, D. 2003. An Extension to PDDL for Expressing Temporal Planning Domains. *JAIR*, 61-124, 2003.

Halpern, J. Y., and Shoham, Y. 1991. A Propositional Modal Logic of Time Intervals. *Journal of the ACM*, 38(4):935-962, 1991.

Halpern, J. Y., and Vardi, M. 1989. The complexity of reasoning about knowledge and time. I. Lower bounds. *Journal of Computer and System Science*, 38:195-237, 1989.

Hoffmann, J. and Edelkamp, S. 2005. The deterministic part of IPC-4: An overview. *Journal of Artificial Intelligence Research*, 24, 519-579, 2005

Kabanza, F. and Thiebaux, S. 2005. Search Control in Planning for Temporally Extended Goals. *ICAPS*, 2005.

Koymans, R. 1990. Specifying real-time properties with metric temporal logic. *Real-Time Systems*, 2(4):255-299, 1990.

McDermott, D. 2004. The Opt and Optop API. Tech. rep., Yale University, 2004.

McDermott, D. and the AIPS-98 Planning Competition Committee. 1998. PDDL-the planning domain definition language. Tech. rep. at: www.cs.yale.edu/homes/dvm, 1998.

Monica, D., Goranko, V., Montanari, A. 2011. Expressiveness of the Interval Logics of Allen's Relations on the Class of all Linear Orders: Complete Classification. In *IJCAI*, 2011.

Monica, D., Goranko, V., Montanari, A., and Sciavicco, G. 2011. Interval Temporal Logics: A Journey. In *Bulletin of the EATCS*, 105, 73-99, 2011.

Molineaux, M., Klenk, M., and Aha, D. W. 2010. Planning in Dynamic Environments: Extending HTNs with Nonlinear Continuous Effects. In *Proceedings of 24th AAAI Conf.*, 2010.

Patrizi, F., Lipoveztky, N., Giacomo, G., and Geffner, H. 2011. Computing Infinite Plans for LTL Goals Using a Classical Planner. *IJCAI*, 2011.

Patrizi, F., Lipoveztky, N., and Geffner, H. 2013. Fair LTL Synthesis for Non-Deterministic Systems using Strong Cyclic Planners. *IJCAI*, 2013.

Penberthy, J., and Weld, D. 1994. Temporal planning with continuous change. In *Proceedings of AAAI Conf.*, 1994.

Rajan,K., Py, F., and Barreiro, J. 2013. Towards Deliberative Control in Marine Robotics. In *M. L. Seto (Ed.), Marine Robot Autonomy*, (pp. 91-175). Springer New York, 2013.

Reynolds, M. 2010. The complexity of temporal logic over the reals. *Annals of Pure & Applied Logic*, 161:1063-1096, 2010.

Sciavicco, G. 2012. Reasoning with Time Intervals: A Logical and Computational Perspective. *ISRN Artif. Intelligence*, 2012.

Smith, D., Frank, J., and Cushing, W. 2008. The ANML Language. The *ICAPS-08 Workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*, 2008.