Specifying Timed Patterns using Temporal Logic

Author(s) omitted for submission

ABSTRACT

Monitoring system behaviors using formal specifications appears to be an effective technique in analyzing cyber-physical systems. However, to achieve intended results in monitoring, specification languages need to be intuitive, elegant, and expressive at the first place. In this paper, we propose a metric extension of well-known Halpern-Shoham (HS) logic, called Metric Compass Logic (MCL), for monitoring purposes. Originally proposed for high-level temporal reasoning, the logic HS is very expressive and enables users to specify many temporal patterns in an intuitive and elegant way. As our main contribution, we present an offline monitoring technique for timed patterns specified in MCL. Our solution is built upon the framework developed for timed regular expressions (TRE) matching but explores a different (logical) direction. We finally study several practical features concerning atomic formulas and discuss a combined timed pattern specification language with TRE.

CCS CONCEPTS

•Computer systems organization → Embedded systems; *Redundancy;* Robotics; **•Networks** → Network reliability;

KEYWORDS

ACM proceedings, LaTeX, text tagging

ACM Reference format:
Author(s) omitted for submission. 2018. Specifying Timed Patterns using Temporal Logic. In *Proceedings of Hybrid Systems: Computation and Control, Porto, Portugal, April 11–13 (HSCC 2018),* 10 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The analysis of complex systems is concerned with behaviors that systems exhibit in response to the environment as well as their properties that cause these behaviors. System behaviors are inherently related to time and can be expressed in a sequential form on the time axis under different names such as signals, waveforms, time series, or event sequences. Observing and evaluating such temporal behaviors triggered by certain external inputs is an essential task for understanding and assessing such systems. We use the term *monitoring* to denote the process of observing and evaluating temporal behaviors of systems.

Monitoring has been always prior to more sophisticated techniques that involve modeling a system mathematically. However, the lack of systematic and rigorous approach prevents its adoption as a reliable tool when reasoning about systems. Following the success of formal methods, especially of (formal) verification of hardware and software systems, there is a growing trend in academia and industry to monitor system behaviors against formal specifications. This approach combines the simplicity of monitoring with the rigor of the formalism without necessitating a mathematical model of the system as in verification. Therefore, monitoring using formal specifications is especially attractive for cyber-physical systems where no model, if exists, is small and accurate enough to be formally verified.

There are several considerations when choosing or designing a formal specification language for monitoring [12, 16, 17]. The historical direction has been to use the same specification language used in verification for monitoring purposes since monitoring using formal specifications was initially considered to be a lightweight verification technique. Therefore, temporal logics with their mostly intuitive (temporal) operators are used for monitoring purposes. Secondly, a highly expressive language is desired for specification. Unfortunately, the complexity of verification (validity) problem poses a significant limit on the expressiveness that a language can reach. It is common that an intuitive and expressive language is intractable for verification purposes. On the other hand, monitoring is a simpler problem complexity-wise and therefore allows the use of much more expressive languages. At this point, we can convince ourselves that specification languages for monitoring can go beyond languages proposed for verification.

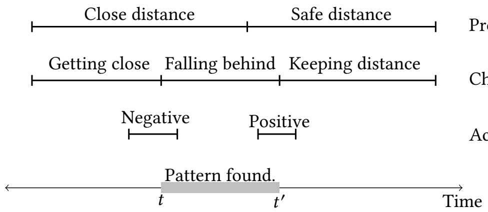

Figure 1: An example.

In this paper, we propose the modal logic of time periods known as Halpern-Shoham (HS) logic [11] and its metric extension named as Metric Compass Logic (MCL) for monitoring purposes. The HS logic is very expressive and it enables users to specify some complex patterns such as a pattern that either *precedes*, *meets*, *begins*, *ends*, *overlaps*, or occurs *during* another pattern. Then, we extend timed pattern matching [27] for temporal logic patterns specified in the logic HS and MCL. Timed pattern matching is a monitoring task defined to be an act of identifying the segments of discrete-valued continuous-time behaviors (called timed behaviors) that satisfy a timed pattern. As a specification language, it has initially employed timed regular expressions (TRE) [2, 3], which augments regular expressions with timing (duration) constraints. Further related works include timed automata patterns [31], online procedures [28, 32], and a software implementation [26]. We now illustrate timed pattern matching using MCL with an example.

Suppose that we collect data from acceleration and proximity sensors attached to some automobiles during the driving. In Figure 1, we depict an example timed behavior obtained from such a car. In this example, the car is getting closer to another vehicle and the (human or automatic) controller reacts to maintain the safe trailing distance. It is usually the case that Over such behaviors, we can specify a query *all falling behind periods begun by a deceleration period and followed by a period of safe and keeping distance at least 30 seconds.* It is not possible to express this pattern using TRE but we can use MCL to specify and extract the query from the driving log. The output would be a set of time periods and indicates instances of the pattern such as a match at (t, t') shown in Figure 1. Clearly these matches can be subject to further analyses and higher-level reasoning tasks. Such a formalization may also help address some legal and liability issues that appear in the context of autonomous driving. For the rest of paper, however, we treat all these concepts in an abstract manner without referring to any particular application.

The paper is structured as follows. Section 2 and 3 give precise definitions of time periods, their representations, and timed behaviors. Our design choices and setup are also explained in detail since all these concepts are floating around with small but important variations. In Section 4, we first review temporal logics based on time periods and focus on HS logic in particular. Then, we introduce metric compass logic and present the syntax and (relational) semantics of MCL. In Section 5, we are interested in the problem of timed pattern matching for MCL over timed behaviors. For that, we formulate an alternative (algebraic) semantics for MCL that allows the evaluation of a formula inductively and show that relational and algebraic semantics of MCL agree. We then describe our implementation and present our experimental results. In Section 6 we give more details on atomic formulas where we have a great deal of freedom that can be tailored for specific applications. Finally we discuss a possible integration with TRE and our future insights in the conclusion.

2 TIME PERIODS & REPRESENTATIONS

For our monitoring purposes, we are interested in dense, linear, and bounded time models. We fix our time domain $T = (0, d)$ to be an interval of reals throughout the paper without loss of generality. We then define time periods over T as follows.

Definition 2.1 (Time period). A time period (t, t') is a pair of begin and end boundaries on a time domain T such that $t < t'$ with a (non-zero) duration of $t' - t$. By $\Omega(T)$, we denote the set $\{(t, t') \mid t \geq 0,\ t' \leq d,\ t' - t > 0\}$ of all time periods over T.

We say that a time period (t_1, t_1') meets another period (t_2, t_2') if the end of the first equals to the begin of the second such that $t_1' = t_2$. A sequence $S = (t_0, t_1), (t_1, t_2), \ldots, (t_{n-1}, t_n)$ of meeting time periods is called a time period sequence. It begins at t_0, ends at t_n and has a duration of $t_n - t_0$ and a (logical) length of n.

Geometrically speaking, a time period (t, t') can be viewed as a line segment on the time axis, or alternatively, as a point

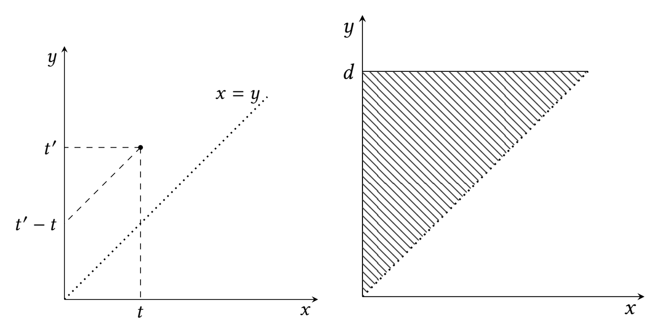

Figure 2: Geometric representations of a time period (t, t') and the set $\Omega(T)$ of all time periods where $T = (0, d)$.

on a two-dimensional xy-plane under standard conventions. In Figure 2-(left), we illustrate attributes of a time period (t, t'), its begin t, its end t', and its duration $t' - t$ on such a plane. Then we are interested in a very specific set of linear inequalities that corresponds to constraints on beginnings, endings, and durations of time periods, often called vertical, horizontal, and diagonal half-planes. In particular, the set $\Omega(T)$ of all time periods forms a triangular set of points on the plane as depicted in Figure 2-(right).

In the following we closely relate sets of time periods to Boolean functions. Boolean (set-theoretic) operations of union ($\cup$), intersection ($\cap$), and complementation ($\overline{\ }$) as well as the inclusion relation ($\subseteq$) over sets of time periods are defined as usual with the empty set $\emptyset$ and the universal set $\Omega(T)$. We call a set of time periods a *timed relation* if and only if it can be expressed as a finite boolean formula over vertical, horizontal, and diagonal half-planes as follows.

Definition 2.2 (Timed Relation). A timed relation $Z \subseteq \Omega(T)$ is a (possibly uncountable) set of time periods that can be represented on the xy-plane by a boolean combination of finitely many half-planes having one of six forms (1) $x \prec c$, (2) $y \prec c$, (3) $y - x \prec c$, (4) $c \prec y - x$, (5) $c \prec y$, and (6) $c \prec x$ where $\prec \in \{<, \leq\}$ and $c \in \mathbb{R}$.

It is clear that the empty set, $\Omega(T)$, and all finite sets of time periods are timed relations. Next we say that a convex timed relation is a timed relation that can be formed only by intersections. We denote by $\mathcal{Z}$ and $\mathcal{Z}^\cap$ the set of all timed relations and the set of all convex timed relations over $\Omega(T)$, respectively. We distinguish six types of half-planes in the definition by superscripted numbers $(1 - 6)$ such as h^1. The complement of an open [closed] half-plane h^k is a closed [open] half-plane h^{7-k} with the same constant c. Intersections or unions of n half-planes $h^k_1, \ldots, h^k_n$ of the same type would be implied by one of the half-planes h^k_i, $i \in 1, \ldots, n$. Therefore, every convex timed relation $z \in \mathcal{Z}^\cap$ can be formed by an intersection of six half-planes $h^1, h^2, h^3, h^4, h^5, h^6$ of each type such that

$$\bigcap_{k=1\ldots6} h^k = \{(x, y) \mid c_6 \prec x \prec c_1 \cap c_5 \prec y \prec c_2 \cap c_4 \prec y-x \prec c_3\}$$

where $c_1, \ldots, c_6 \in \mathbb{R} \cup \{-\infty, +\infty\}$. Consequently we can represent a convex timed relation as a six-tuple $(h^1, h^2, h^3, h^4, h^5, h^6)$ of half-spaces. Notice that these half-planes are not totally independent of each other and an arithmetic combination of two other may imply a tighter half-plane. Clearly every non-empty convex timed relation $z \in \mathcal{Z}^\cap$ has a unique normal representation such that all half-planes are tight. More precisely, given a representation $(h^1, h^2, h^3, h^4, h^5, h^6)$ of z that the tight representation of z can be found as follows:

$$\begin{aligned}\textsc{tighten}(z) = \Big(& h^1 \cap (h^2 + h^4),\\ & h^2 \cap (h^1 + h^3),\\ & h^3 \cap (h^2 + h^6),\\ & h^4 \cap (h^1 + h^5),\\ & h^5 \cap (h^4 + h^6),\\ & h^6 \cap (h^3 + h^5)\Big)\end{aligned}$$

where + denotes arithmetic addition of inequalities. In Figure 3 we illustrate the most general (hexagon) case for a convex timed relation where you can see that two bounds $h^1 : x < c_1$ and $h^3 : y - x \leq c_3$ imply a bound $h^1 + h^3 : y < c_1 + c_3$, which can absorb or be absorbed by h^2. Importantly, an inclusion test between two convex timed relations z_1 and z_2 can be performed over their tight representations such that

$$z_1 \subseteq z_2 \longleftrightarrow \bigwedge_{i=1,\ldots,6} h^i_1 \subseteq h^i_2$$

From now on, we consider all representations of convex timed relations to be tightened according to the definition above and we use the term *zone* both for a convex timed relation and its tight representation. And we say a zone z_1 is implied by another zone z_2 if $z_1 \subseteq z_2$.

We represent a timed relation as a finite union of zones similar to disjunctive normal forms of Boolean functions. By definition and by DeMorgan's laws, every timed relation Z can be representable by a union over a finite set of non-empty zones $R_Z = \{z_1, z_2, \ldots, z_n\}$ such that $Z = z_1 \cup z_2 \cup \cdots \cup z_n$. Colloquially we say a timed relation when we want to emphasize semantic aspects whereas a union of zones to emphasize syntactic aspects.

We say that a union of zones R_Z is absorptive if and only if no zone in R_Z is implied by any other zone in R_Z. For any union of zones R_Z, we can obtain an equivalent absorptive union of zones, denoted by $\textsc{absorb}(R_Z)$, by removing all

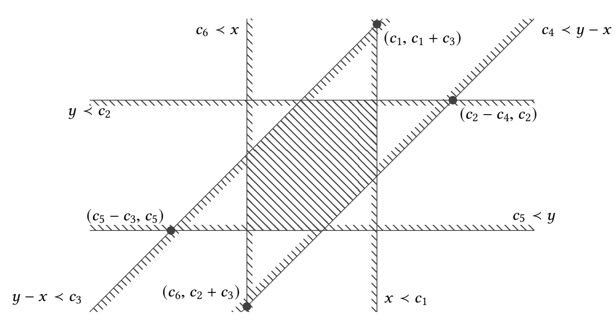

Figure 3: Intra-dependencies between half-planes of a convex timed relation.

absorbed zones from the representation. Obviously, there may be different representations of a timed relation Z but an absorptive representation ABSORB(R_Z) for a given R_Z is unique. Unless specified otherwise, we consider all unions of zones to be absorptive for the rest.

Finally we review Boolean operations on such representations. The union $R_{Z_1} \cup R_{Z_2}$ is simply the union of zones from both representation. The intersection $R_{Z_1} \cap R_{Z_2}$ is computed as a pairwise intersection between member zones. These operations are typically followed by an ABSORB operation, which also has a quadratic worst-case complexity. For complementation, since both DeMorgan's laws hold for timed relations, we can complement a zone z as follows.

$$\overline{R_z} = \{\overline{h_z^1},\ \overline{h_z^2},\ \overline{h_z^3},\ \overline{h_z^4},\ \overline{h_z^5},\ \overline{h_z^6}\}$$

And then a complement union of zones $\overline{R_Z}$ can be developed into a union of zones in an incremental and efficient manner such that

$$\overline{R_Z^i} = \text{ABSORB}\left(\overline{R_Z^{i-1}} \ \cap \ \overline{R_{z_i}}\right)$$

where $\overline{R_Z^i}$ is the complement of the subset $\{z_1, z_2, \ldots, z_i\}$ of a union of zones with n elements for $1 \leq i \leq n$ and $\overline{R_Z^0} = \Omega(T)$. The current upper bound for the worst-case complexity of complementation is $O(n^4)$, which is quite pessimistic.

Note that zones (possibly in higher dimensions) are commonly employed for timed systems research and admit efficient data structures called difference bound matrices (DBM) [6, 9]. Here we use many two-dimensional zones to represent time periods as proposed in [27] rather than one or a few high-dimensional zones to represent clock valuations.

3 TIMED BEHAVIORS

Let $P = \{p_1, \ldots, p_m\}$ be a finite set of (atomic) propositional variables over a time domain T that correspond to some states or activities of some real-time systems and the environment. Examples include the proposition *turning left* denoting an activity of a car and *the engine temperature is higher than 110 degrees* which denotes a qualitative state over a physical quantity. We consider such propositions to be observed continuously for a finite amount of time and model the evolution of corresponding activities and states as timed behaviors. For that we first define an alphabet $\Sigma = \mathbb{B}^m$ of observations expressed as Boolean vectors of dimension m where $\mathbb{B} = \{0, 1\}$ and $a(i)$ is the value of p_i for $i = 1 \ldots m$. Then a timed behavior is a finite sequence of time periods such that each period (t_{k-1}, t_k) is associated with a Boolean vector $a_k \in \Sigma$ such that $a_k(i) = 1$ if the proposition p_i holds on the time period (t_{k-1}, t_k) and $a_k(i) = 0$ if it does not.

Definition 3.1 (Timed Behavior). A timed behavior w over a set of atomic propositions P on a time period (t_0, t_n) is a non-empty finite sequence such that

$$w = (t_0, t_1, a_1), (t_1, t_2, a_2), \ldots, (t_{n-1}, t_n, a_n)$$

where $a_k \in \Sigma$ and $t_{k-1} < t_k$ for $k \in 1 \ldots n$. A timed behavior begins at t_0, ends at t_n, has a duration of $t_n - t_0$ and a (logical) length of n.

We use $w_p(t, t')$ to denote the restriction of w to an atomic proposition p and a time period (t, t'). The concatenation $w_1 \cdot w_2$ is defined only if w_1 meets w_2, that is, w_1 ends at the point where w_2 begins. Alternatively, a timed behavior can be given as a sequence of pairs of duration values and symbols from the alphabet Σ such that

$$w = (t_1 - t_0, a_1), (t_2 - t_1, a_2), \ldots, (t_n - t_{n-1}, a_n)$$

In this notation, we assume the beginning time $t_0 = 0$ unless otherwise stated. As an example, we depict in Figure 4 an evolution of two atomic propositions $P = \{p_1, p_2\}$ over time where the alphabet is $\Sigma = \{(0,0), (0,1), (1,0), (1,1)\}$. Then a timed behavior w can represent this evolution as

$$w = (0, 2, (1,1)), (2, 4, (1,0)), (4, 6, (0,0)), (6, 7, (0,1)), (7, 8, (1,1)), (8, 9, (0,1)), (9, 10, (1,1)), (10, 11, (1,0)), (11, 12, (0,0))$$

or, using duration-symbol notation,

$$w = (2, (1,1)), (2, (1,0)), (2, (0,0)), (1, (0,1)), (1, (1,1)), (1, (0,1)), (1, (1,1)), (1, (1,0)), (1, (0,0))$$

Clearly such representations are not unique; any time period associated with a letter (observation) can be divided into shorter periods of the same letter and the resulting behaviors would be equivalent. Sometimes these successive periods of the same letter are called *stuttering* periods [3, 15]. Stuttering periods are often avoidable and it is more efficient to work with stutter-free behaviors after merging periods. For example, if we restrict w above to the proposition p_2, we can directly write w_{p_2} as

$$w_{p_2} = (0, 2, (1)), (2, 4, (0)), (4, 6, (0)), (6, 7, (1)), (7, 8, (1)), (8, 9, (1)), (9, 10, (1)), (10, 11, (0)), (11, 12, (0))$$

which, after elimination of stuttering, becomes

$$w_{p_2} = (0, 2, (1)), (2, 6, (0)), (6, 10, (1)), (10, 12, (0))$$

However, there are some applications such as online monitoring where you do not know what will be observed in the next period but want to act immediately; for such cases,

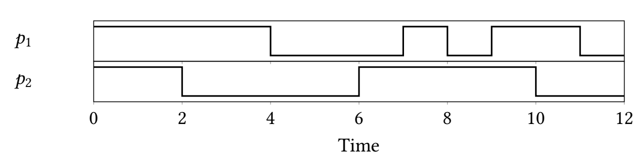

Figure 4: A behavior of atomic propositions p_1 and p_2.

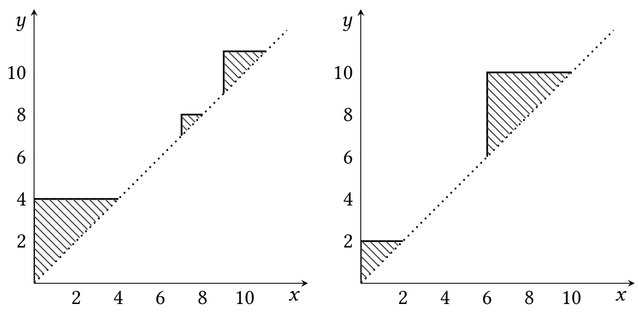

Figure 5: Valuations of atomic propositions.

stuttering in behaviors may arise naturally. Therefore, we (have to) allow stuttering in our definitions and procedures, and the correctness of our results should not depend on stutter-freeness.

Next we define our period-based temporal structures that we use to evaluate propositions over timed behaviors. Given a timed behavior w over a set P of propositions on a time domain T, we define a temporal structure $W = (\Omega(T), V)$ induced by w such that $\Omega(T)$ is the set of all time periods over T and $V : P \longrightarrow 2^{\Omega(T)}$ is a valuation function that assigns every proposition to a set of time periods on which it holds with respect to w.

It is an important point that atomic propositions are homogeneous, that is to say, an atomic proposition $p \in P$ holds on a time period if and only if it holds on its all sub-periods, which is captured by the formula

$$(t, t') \in V(p) \longleftrightarrow \forall r, r'.\ (t < r < r' < t') \rightarrow (r, r') \in V(p)$$

Since a timed behavior w is a finite sequence, a valuation $V(p)$ of an atomic proposition p in W is a finite union of triangular zones that reside along the diagonal, thus a timed relation. We illustrate $V(p_1)$ and $V(p_2)$ in Figure 5 for the timed behavior in Figure 4.

4 LOGICS OF TIME PERIODS

In this section, we study temporal logics based on time periods (rather than time points) for monitoring purposes. By temporal logic, we mean the modal approach introduced by Prior in [22, 23] under the name of tense logic with two temporal modalities, namely *sometime in the past* (P) and *sometime in the future* (F). These modalities are based on time points and easily seen that they implicitly refer to time points *less than* the current time point and time points *greater than* the current time point, respectively. Later Pnueli imported the concept into the computer science and proposed the use of temporal logic to specify infinite-duration temporal properties [21]. Afterwards, temporal logic quickly was adopted by the community as a major specification formalism in modern verification technology.

However, point-based temporal logics are not naturally suitable to express local patterns with finite durations. For these specifications, temporal logics based on time periods are argued to be a better formalism. Then, considering time periods as primitive entities, Allen introduced 13 basic relations between two time periods to represent high-level temporal knowledge in [1]. The set of so-called Allen's relations consists of relations *met-by* (A), *begins* (B), *ends* (E), *during* (D), *overlaps* (O), and *later* (L) as well as their inverses and the equality (=). In Figure 6 we illustrate these relations in a way that a depicted time period (horizontal lines) and the time period (t, t') is in the specified relation given at the right. In [11], Halpern and Shoham applied Prior's modal approach over time periods and proposed a temporal logic that features a modality for each Allen's relation.

It is shown that six certain temporal modalities of the HS logic can express others under strict semantics. Here we use the compass notation introduced by Venema in [29] (with our slight extension) since it has nice geometric connotations on the two-dimensional plane. The basic set consists of six modalities (diamonds) denoted by $\Diamond, \Diamond, \Diamond, \Diamond, \Diamond, \Diamond$, respectively corresponding to relations $A, A^{-1}, B, B^{-1}, E, E^{-1}$ between time periods. These (diamond) modalities have the following intuitive meanings over a formula φ over time periods.

$\Diamond\ \varphi$	φ holds at a period met by the current one.
$\Diamond\ \varphi$	φ holds at a period that meets the current one.
$\Diamond\ \varphi$	φ holds at a period that begins the current one.
$\Diamond\ \varphi$	φ holds at a period begun by the current one.
$\Diamond\ \varphi$	φ holds at a period that ends the current one.
$\Diamond\ \varphi$	φ holds at a period ended by the current one.

As shown, using these basic set of modalities, we can derive more modalities for the remaining relations. For example, $\Diamond\varphi \equiv \Diamond\Diamond\varphi$ corresponds to the relation L, $\Diamond\varphi \equiv \Diamond\Diamond\varphi \equiv \Diamond\Diamond\varphi$ to the relation D, and $\Diamond\varphi \equiv \Diamond\Diamond\varphi$ to the relation O. Moreover, the dual (box) modalities are defined as usual such that $\Box\varphi = \overline{\Diamond\overline{\varphi}}$ where $\Diamond \in \{\Diamond, \Diamond, \Diamond, \Diamond, \Diamond, \Diamond\}$ and $\Box$

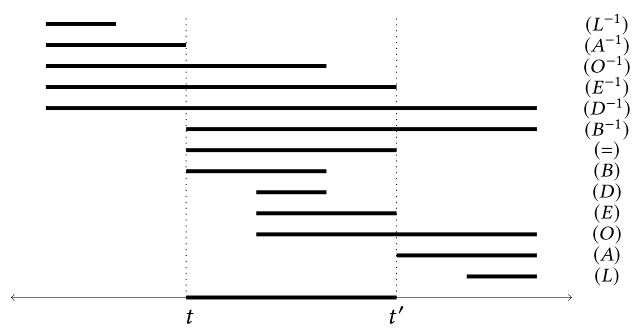

Figure 6: 13 basic relations between time periods.

matches the decoration. From another point of view, each modality accesses a different region on the two-dimensional plane with respect to the current period and quantifies φ over the accessed region. In Figure 7, we illustrate accessed regions for each Allen's relation with respect to a period (t, t').

Finally, we mention the chop modality, which corresponds to the concatenation operator of regular expressions and can be added on the top of temporal logics [20, 25, 30]. It is known that the expressiveness of HS logic can be increased further by the addition of the chop modality [30]. We discuss the chop in more detail in the conclusion.

We now introduce metric compass logic (MCL) as a metric extension of the HS logic. Metric compass logic augments temporal modalities of the HS logic with temporal constraints similar to the way that metric temporal logic (MTL) [14] extends linear temporal logic [10, 21]. Note that we slightly differ from the original definitions by excluding degenerate (zero-duration) time periods and employing irreflexive (strict) versions of modalities.

Among fragments of HS logic, propositional neighborhood logic (PNL) [19], which only includes modalities for meets (A^{-1}) and met-by (A) relations of Allen, is studied with metric constraints [7]. We are not aware of a more expressive metric extension in the literature. It is perhaps the case that undecidability results (of the validity problem) prevented any further proposal on metric extensions. However, since we propose metric compass logic for monitoring purposes, we are only interested in evaluating a MCL formula for a given timed behavior in this paper. Therefore, we do not have any reason to restrict ourselves to a fragment and we extend the HS system in its full generality in the following.

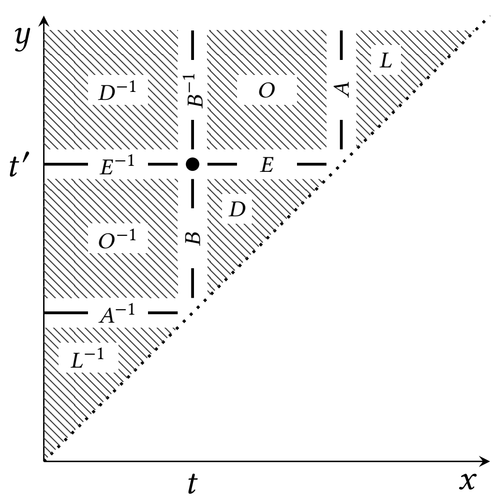

Figure 7: Accessed regions for each Allen relation with respect to a time period (t, t').

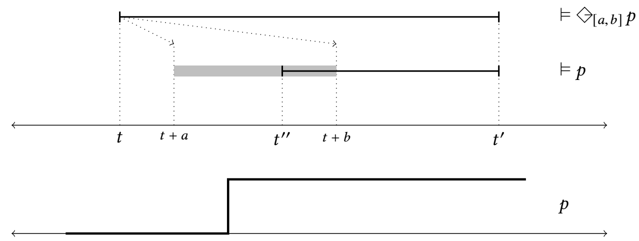

Figure 8: One dimensional illustration of $\Diamond_{[a,b]}$.

The syntax of metric compass logic that admits usual Boolean connectives and metric compass modalities is given by the following grammar:

$$\varphi \;:=\; p \mid \overline{\varphi} \mid \varphi_1 \cup \varphi_2 \mid \varphi_1 \cap \varphi_2 \mid \Diamond_I\, \varphi$$

where $p \in P$ is a propositional variable, $\Diamond \in \{\Diamond, \Diamond, \Diamond, \Diamond, \Diamond, \Diamond\}$ is a metric (time-bounded) compass modality, and $I \subseteq \mathbb{R}_{>0}$ is an interval of non-zero duration values. We omit the interval I if $I = (0, \infty)$. Given a timed behavior w, the satisfaction of a metric compass logic formula for a time period (t, t') is defined inductively.

Definition 4.1 (Relational Semantics). The satisfaction $\models$ of a metric compass logic formula φ in a temporal structure $W = (\Omega(T), V)$ induced by a timed behavior w with a time domain T, relative to a time period $(t, t') \in \Omega(T)$ is defined as follows:

$$\begin{array}{lll}
(\mathcal{W}, t, t') \models p & \leftrightarrow & (t, t') \in V(p) \\
(\mathcal{W}, t, t') \models \overline{\varphi} & \leftrightarrow & (\mathcal{W}, t, t') \not\models \varphi \\
(\mathcal{W}, t, t') \models \varphi_1 \cup \varphi_2 & \leftrightarrow & (\mathcal{W}, t, t') \models \varphi_1 \\
 & & \quad \text{or } (\mathcal{W}, t, t') \models \varphi_2 \\
(\mathcal{W}, t, t') \models \Diamond_I\, \varphi & \leftrightarrow & \exists t'' \in (t, t').\ t' - t'' \in I \\
 & & \quad \text{and } (\mathcal{W}, t, t'') \models \varphi \\
(\mathcal{W}, t, t') \models \Diamond_I\, \varphi & \leftrightarrow & \exists t'' \in (t', \infty).\ t'' - t' \in I \\
 & & \quad \text{and } (\mathcal{W}, t, t'') \models \varphi \\
(\mathcal{W}, t, t') \models \Diamond_I\, \varphi & \leftrightarrow & \exists t'' \in (t, t').\ t'' - t \in I \\
 & & \quad \text{and } (\mathcal{W}, t'', t') \models \varphi \\
(\mathcal{W}, t, t') \models \Diamond_I\, \varphi & \leftrightarrow & \exists t'' \in (-\infty, t).\ t - t'' \in I \\
 & & \quad \text{and } (\mathcal{W}, t'', t') \models \varphi \\
(\mathcal{W}, t, t') \models \Diamond_I\, \varphi & \leftrightarrow & \exists t'' \in (t', \infty).\ t'' - t' \in I \\
 & & \quad \text{and } (\mathcal{W}, t', t'') \models \varphi \\
(\mathcal{W}, t, t') \models \Diamond_I\, \varphi & \leftrightarrow & \exists t'' \in (-\infty, t).\ t - t'' \in I \\
 & & \quad \text{and } (\mathcal{W}, t'', t) \models \varphi
\end{array}$$

Each MCL modality $\Diamond_I$ above fixes one dimension (either t or t') in their semantics, and the metric constraint I restricts the range of quantification over the other (free) dimension. For example, the modality $\Diamond_{[a,b]}$ fixes the endpoint t' and quantifies over a restricted range $[t + a, t + b]$ as illustrated in Figure 8. Note that $\Diamond$ can be seen as a two-dimensional analog of Prior's *sometime in the future* (F) if you consider endpoints of periods to be fixed at the infinity and the reasoning is essentially performed over (begin) points.

5 OFFLINE MATCHING

In this section, we show how to match timed patterns specified in metric compass logic. Pattern matching is considered to be a computation for finding and reporting all satisfying segments, called matches, of an input sequence that satisfy a predefined pattern. The set of all satisfying segments is called the match set of the pattern over a timed behavior. Assuming the input behavior w and valuations of propositions is completely available before matching, computing the match set is equivalent to evaluating φ inductively in the temporal structure induced by w.

It is known that there is a very close connection between modal logics and boolean algebras with operators (BAO) [13] studied in several monographs [4, 5]. Such algebras provide alternative (algebraic) semantics for various modal logics and can be used to evaluate a formula. For that, Boolean connectives and modalities are interpreted as operations on valuations of propositions. In the following, we follow a similar route for MCL and start by defining algebraic semantics of MCL before presenting metric compass operators on timed relations.

Definition 5.1 (Algebraic Semantics). For a temporal structure $\mathcal{W} = (\Omega(T), V)$, we extend the valuation function V for arbitrary MCL formulas as follows:

$$\begin{array}{rcl} V(p) & = & V(p) \text{ for } p \in P \\ V(\overline{\varphi}) & = & \Omega(T) \setminus V(\varphi) \\ V(\varphi_1 \cup \varphi_2) & = & V(\varphi_1) \cup V(\varphi_2) \\ V(\varphi_1 \cap \varphi_2) & = & V(\varphi_1) \cap V(\varphi_2) \\ V(\Diamond_I \varphi) & = & \Diamond_I V(\varphi) \end{array}$$

From Section 2 and 3, we know that valuations of atomic propositions are timed relations as well as timed relations are closed under Boolean operations. We now define metric compass operations on timed relations and show that timed relations are closed under these operations. A metric compass modality $\Diamond_I$ is interpreted as an operator on a timed relation Z as follows.

$$\begin{array}{rcl} \Diamond_I Z & = & \{(x,y) \mid \exists r.\ x < r < y,\ y - r \in I, \text{ and } (x,r) \in Z\} \\ \Diamond_I Z & = & \{(x,y) \mid \exists r.\ x < y < r,\ r - y \in I, \text{ and } (x,r) \in Z\} \\ \Diamond_I Z & = & \{(x,y) \mid \exists r.\ x < r < y,\ r - x \in I, \text{ and } (r,y) \in Z\} \\ \Diamond_I Z & = & \{(x,y) \mid \exists r.\ r < x < y,\ x - r \in I, \text{ and } (r,y) \in Z\} \\ \Diamond_I Z & = & \{(x,y) \mid \exists r.\ x < y < r,\ r - y \in I, \text{ and } (y,r) \in Z\} \\ \Diamond_I Z & = & \{(x,y) \mid \exists r.\ r < x < y,\ x - r \in I, \text{ and } (r,x) \in Z\} \end{array}$$

In Figure 9 we illustrate each metric compass (diamond) operator over a timed relation that contains a single time period $(1,2)$. Intuitively speaking, a diamond operator shifts a time period in the specific direction on the plane by an allowed amount. This technique, called back-shifting (of time points), is used to evaluate the timed eventually modality of metric temporal logic [18]. Notice that the shift of time points can be viewed as a degenerate case of that of timed periods and there are surely more directions to move in two dimensions. Next we note that metric compass operators possess two important algebraic properties of normality and additivity such that

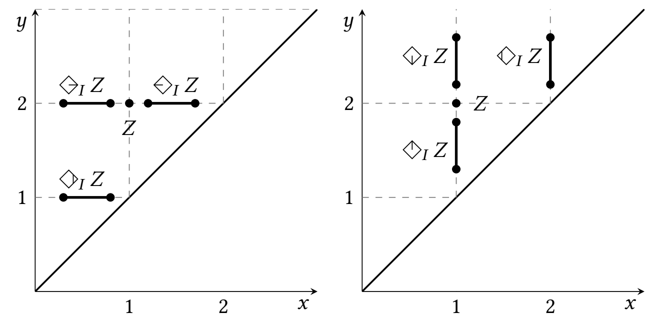

Figure 9: Metric compass operations $\Diamond_I$ on a timed relation $Z = \{(1,2)\}$ for a temporal constraint $I = [0.2, 0, 7]$.

$$\Diamond_I \emptyset = \emptyset$$

$$\Diamond_I(Z_1 \cup Z_2) = \Diamond_I Z_1 \cup \Diamond_I Z_2$$

Therefore, the algebra of timed relations with (metric) compass operators forms a boolean algebra with operators in the sense of [13].

In Proposition 5.2, we finally show that algebraic and relational semantics of metric compass logic agree.

PROPOSITION 5.2 (SEMANTIC AGREEMENT). *For every temporal structure $\mathcal{W} = (\Omega(T), V)$ and every MCL formula, the statement $(\mathcal{W}, t, t') \models \varphi \leftrightarrow (t, t') \in V(\varphi)$ holds.*

PROOF. By induction on the structure. Cases for propositions and Boolean operations are straightforward. Then we only show the case $\Diamond$ as cases for other compass operators are symmetric. We directly show

$$\begin{aligned} (\mathcal{W}, t, t') \models \Diamond_I \varphi &\leftrightarrow (t, t') \in V(\Diamond_I \varphi) \\ &\leftrightarrow \exists t''.\ t < t'' < t',\ t'' - t \in I, \\ &\qquad \text{and } (t'', t') \in V(\varphi) \\ &\leftrightarrow \exists t''.\ t < t'' < t',\ t'' - t \in I, \\ &\qquad \text{and } (\mathcal{W}, t'', t') \models \varphi \end{aligned}$$

□

In the following we show the class of zones is also closed under metric compass operators. Consequently we have that applying a metric compass operation on a zone results in a zone whose bounds are shifted according to the type of compass operation and the metric constraint.

Proposition 5.3. *Given a zone z, the timed relation $\Diamond_I z$ is a zone.*

Proof. We only show the case of $\Diamond_{[m,n]}$ as other cases are symmetric. Following the semantics of $\Diamond_{[m,n]}$, we have $(x, y) \in \Diamond_{[m,n]}$ iff $\exists r \in (x, y).\ r - x \in [m, n]$, and $(r, y) \in z$ which translates to $\exists r \in (x, y)$ such that

$$\left\{\begin{array}{ccccc} \underline{b} & \prec & r & \prec & \overline{b} \\ \underline{e} & \prec & y & \prec & \overline{e} \\ \underline{d} & \prec & y-r & \prec & \overline{d} \end{array}\right\}$$

By eliminating the quantifier, we obtain that $\Diamond_{[m,n]}$ equals to a zone

$$\left\{\begin{array}{ccccc} \underline{b}-n & \prec & x & \prec & \overline{b}-m \\ \underline{e} & \prec & y & \prec & \overline{e} \\ \underline{d}+m & \prec & y-x & \prec & \overline{d}+n \end{array}\right\}$$

□

For example, consider the zone $z = \{(x, y) \mid 3 \le x \le 5 \cap 5 \le y \le 7 \cap 3 \le y - x \le 5\}$ at left of Figure 10. Then we have $\Diamond_{[1,2]} z = \{(x, y) \mid 1 \le x \le 4 \cap 5 \le y \le 7 \cap 4 \le y - x \le 7\}$ at right of the same figure obtained by shifting z towards the left accordingly. Equivalently, this operation can be viewed as a Minkowski sum $z \oplus S_{left}$ of the zone z and a left-shifting set $S_{left} = \{(-t, 0) \mid t \in [1, 2]\}$ with respect to $\Omega(T)$.

Following the proof above, we apply metric compass operations over zones directly and extend it towards unions of zones R_Z as $\Diamond_I R_Z = \{\Diamond_I z \mid z \in R_Z\}$ followed by an absorb operation.

In Algorithm 1, we present an offline matching (evaluation) algorithm $\text{eval}_W(\varphi)$, suggested by the algebraic semantics of metric compass logic. For implementation, we extend the zone library of open-source tool montre [26] with our metric compass operators and complementation on zones. We perform our experiments on a single-core 3.3GHz machine for a set of test patterns that are specified by metric compass logic. Input behaviors are generated by repeating

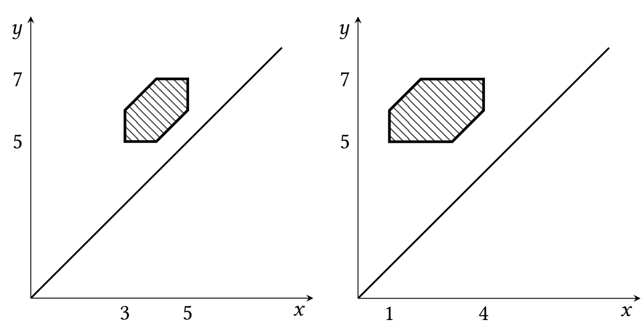

Figure 10: A zone z (left) and its left-shift $\Diamond_{[1,2]} z$ (right).

Algorithm 1 $\text{eval}_W(\varphi)$ with respect to $W = (\Omega(T), V)$

select (φ)
case p:
  $Z := V(p)$
case $\overline{\psi}$:
  $Z :=$ complement($\text{eval}_W(\psi)$)
case $\psi_1 \cup \psi_2$:
  $Z :=$ union($\text{eval}_W(\psi_2)$, $\text{eval}_W(\psi_2)$)
case $\psi_1 \cap \psi_2$:
  $Z :=$ intersect($\text{eval}_W(\psi_2)$, $\text{eval}_W(\psi_2)$)
case $\Diamond_I \psi$:
  $Z := \Diamond$-shift($\text{eval}_W(\psi), I$)
end select
return Z

Test Patterns	Input Size		
	100K	500K	1M
p	0.06/17	0.27/24	0.51/33
$\overline{p}$	0.18/12	0.95/45	1.88/92
$\Diamond_I p$	0.07/16	0.29/65	0.66/163
$\Box_I p$	0.49/23	1.98/100	3.92/163
$\Diamond_I \Diamond_J p$	0.08/20	0.32/37	0.96/60
$\Diamond \Diamond(\Box \varphi)$	0.40/31	1.98/143	3.93/268
$\Diamond \Diamond(\Box \varphi) \cap \Diamond_I q$	0.43/38	2.17/179	4.30/304

Table 1: Execution time/Memory usage (seconds/MBs)

instances of corresponding patterns; thus the number of instances (zones) matched is linear in the size of the behavior. This number in practice is expected to be much smaller than our generated examples as typical in pattern matching. We depict performance results in Table 1 over timed behaviors of length 100K, 500K, and 1M. We use the gnu `time -v` facility to measure execution times (user cpu time) and memory usage (maximum resident set size). For typical cases, experiments suggest a linear time performance with respect to the number of segments in the input behaviors. Notice that complementation is a more costly operation than diamond operators as expected.

6 MORE ON ATOMIC PROPOSITIONS

For our system, we indeed have a great deal of freedom for atomic formulas and their valuations since a valuation function can be any function that returns a timed relation over a timed behavior. In this section, we study some additional features for atomic formulas that enhance the usefulness of timed pattern matching. First, we mention the usual predicate extension as we implicitly used in the motivation of this paper. Then we consider two features regarding atomic formulas, namely Boolean layer and anchors, that are practically implemented by the tool MONTRE [26]. We here formalize these features in our system and show that they are easily expressible in metric compass logic.

Predicates over Real-Valued Behaviors. Atomic propositions can be easily extended towards predicates over real-valued behaviors in the usual sense. Although simple, it is a very useful extension in practice when reasoning about physical observations and hybrid systems in general. The simplest and most common type of predicates are threshold comparisons such as $p(x) : x < c$ and $p(x) : x > c$ where x is a real-valued variable and c is a constant. Consequently we can apply timed pattern matching over real-valued signals via such a symbolic abstraction for the value domain, sometimes called quantization or categorization.

Boolean Layer. Given a set $P = \{p_1, \ldots, p_m\}$ of propositions, atomic expressions are extended towards Boolean expressions over propositions. The syntax of Boolean expressions over P, called the Boolean layer, are defined as

$$\phi := p \mid \neg\phi \mid \phi_1 \wedge \phi_2$$

where $p \in P$ and $\neg$, $\wedge$ are negation and conjunction operators. We derive the disjunction $\phi_1 \vee \phi_2 = \neg(\neg\phi_1 \wedge \neg\phi_2)$ as usual. Intuitively speaking, such a Boolean expression ϕ can be viewed as a new atomic proposition in the system and its truth value, denoted by $\phi(a)$, for $a \in \Sigma$ at each time period are obtained by applying a substitution $\{p_1 \mapsto a(i), \ldots, p_m \mapsto a(m)\}$. More precisely, we characterize Boolean operations on atomic propositions via the valuation function V as follows. Provided that $V(\phi)$, $V(\phi_1)$, $V(\phi_2)$ satisfy the homogeneity property, we have $V(\phi_1 \wedge \phi_2) = V(\phi_1 \cap \phi_2)$ and

$$\begin{aligned} V(\neg\phi) &= \{(t,t') \in \Omega(T) \mid \forall r, r'. \\ &\qquad (t < r < r' < t') \rightarrow (r,r') \notin V(\phi)\} \\ &= V(\overline{\Diamond\Diamond\phi}) \end{aligned}$$

Hence, Boolean layer over atomic formulas is expressible in MCL. In Figure 11, we illustrate valuations of such Boolean expressions over two example propositions p_1 and p_2.

Anchors. We consider rise ($\blacktriangleleft\ \phi$) and fall ($\phi\ \blacktriangleright$) anchor operations over an atomic proposition ϕ and then the rise-fall anchor is derived as $\blacktriangleleft\ \phi\ \blacktriangleright\ = \ \blacktriangleleft\ \phi\ \cap\ \phi\ \blacktriangleright$. These operators are motivated by a practical need to fix begin [end, begin-end] points of matches to rise [fall, rise-fall] points of atomic propositions in timed behaviors. Intuitively speaking, a time period (t, t') of ϕ that begins with a rise point means that there is no meeting period (r, t) that satisfies ϕ. The case for fall point is symmetric. Then, provided that $V(\phi)$

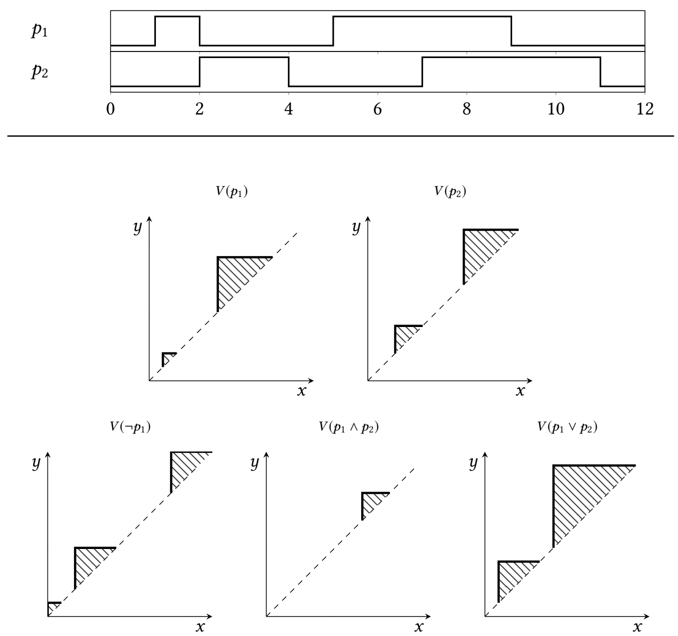

Figure 11: Valuations of Boolean formulas.

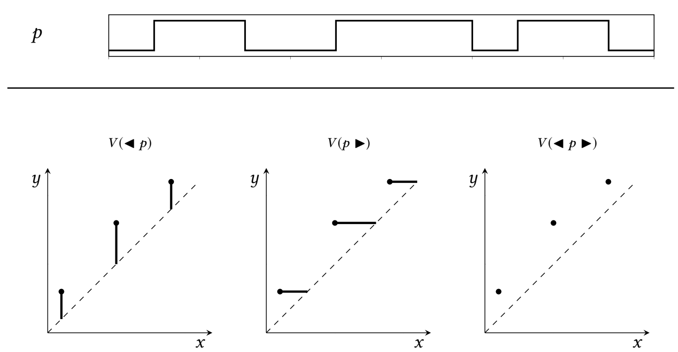

Figure 12: Valuations of anchored formulas.

satisfies the homogeneity property, we have

$$\begin{aligned} V(\blacktriangleleft\ \phi) &= \{(t,t') \in V(\phi) \mid \forall t''. \ (t'', t) \notin V(\phi)\} \\ &= V(\overline{\Diamond\phi} \cap \phi) \end{aligned}$$

$$\begin{aligned} V(\phi\ \blacktriangleright) &= \{(t,t') \in V(\phi) \mid \forall t''. \ (t', t'') \notin V(\phi)\} \\ &= V(\overline{\Diamond\phi} \cap \phi) \end{aligned}$$

Hence, anchor operations over atomic formulas are expressible in MCL. In Figure 12, we illustrate valuations of anchored expressions over an atomic proposition p.

7 CONCLUSIONS

In this paper, we considered the most well-known modal logic of time periods, HS logic and introduced its metric extension MCL. We proposed MCL to be a new timed pattern specification language next to previous proposals of timed regular expressions and timed automata. We provided an offline matching algorithm for MCL patterns and presented our experimental results that suggest promising performance to be explored further.

To our knowledge, this is the first study for pattern matching using a period-based temporal logic specification but also among a few studies for its monitoring in general. The notable work in this context is Interval Temporal Logic (ITL) [20] and its real-time extension Duration Calculus (DC) [8, 24]. The ITL provides the chop (concatenation) and chop-star (Kleene star) modalities over a discrete time axis and is closely related to regular expressions. The HS logic and ITL are incomparable in expressiveness since some HS modalities such as $\Diamond$ are not definable using the chop and the chop is not definable using HS modalities.

The previous work [27] on timed pattern matching provides a framework and algorithms for timed regular expressions (TRE). We have used the same framework to implement our metric compass operators and complementation, which are missing in TRE. Therefore, it is straightforward to combine and even freely mix operators from MCL and TRE for offline matching. Such a combination would result in even more expressive pattern specification language. This also opens several theoretical and practical problems.

We are also interested in the online matching problem for MCL. For that, the first issue is that MCL is not a causal specification language like TRE. Since causality is an important consideration for online monitoring, it may be better to consider in the future a causal fragment of HS, which excludes $\Diamond$ and $\Diamond$.

REFERENCES

[1] James F Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[2] Eugene Asarin, Paul Caspi, and Oded Maler. A Kleene theorem for timed automata. In *Logic in Computer Science (LICS)*, pages 160–171, 1997.

[3] Eugene Asarin, Paul Caspi, and Oded Maler. Timed regular expressions. *Journal of ACM*, 49(2):172–206, 2002.

[4] Patrick Blackburn, Maarten De Rijke, and Yde Venema. *Modal Logic*, volume 53. Cambridge University Press, 2002.

[5] Patrick Blackburn, Johan van Benthem, and Frank Wolter. *Handbook of Modal Logic*. Elsevier, 2006.

[6] Marius Bozga, Susanne Graf, and Laurent Mounier. If-2.0: A validation environment for component-based real-time systems. In *Computer Aided Verification (CAV)*, pages 343–348. Springer, 2002.

[7] Davide Bresolin, Dario Della Monica, Valentin Goranko, Angelo Montanari, and Guido Sciavicco. Metric propositional neighborhood logics: Expressiveness, decidability, and undecidability. In *ECAI*, volume 10, pages 695–700, 2010.

[8] Zhou Chaochen, Charles Anthony Richard Hoare, and Anders P Ravn. A calculus of durations. *Information processing letters*, 40(5):269–276, 1991.

[9] David L Dill. Timing assumptions and verification of finite-state concurrent systems. In *International Conference on Computer Aided Verification*, pages 197–212. Springer, 1989.

[10] Dov Gabbay, Amir Pnueli, Saharon Shelah, and Jonathan Stavi. On the temporal analysis of fairness. In *Proceedings of the 7th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 163–173. ACM, 1980.

[11] Joseph Y Halpern and Yoav Shoham. A propositional modal logic of time intervals. *Journal of the ACM (JACM)*, 38(4):935–962, 1991.

[12] Klaus Havelund and Giles Reger. Runtime verification logics a language design perspective. In *Models, Algorithms, Logics and Tools*, pages 310–338. Springer, 2017.

[13] Bjarni Jonnson and Alfred Tarski. Boolean algebras with operators. *American Journal of Mathematics*, 74(1):127–162, 1952.

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

[15] Leslie Lamport. What good is temporal logic? In *IFIP congress*, volume 83, pages 657–668, 1983.

[16] Martin Leucker and Christian Schallhart. A brief account of runtime verification. *J. Log. Algebr. Program.*, 78(5):293–303, 2009.

[17] Oded Maler. Some thoughts on runtime verification. In *International Conference on Runtime Verification*, pages 3–14, 2016.

[18] Oded Maler and Dejan Nickovic. Monitoring temporal properties of continuous signals. In *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*, pages 152–166. Springer, 2004.

[19] Angelo Montanari and Guido Sciavicco. A decidable logic for time intervals: Propositional neighborhood logic. In *Proc. of the AAAI-2002 Workshop on Spatial and Temporal Reasoning*, pages 27–34, 2002.

[20] Ben Moszkowski. Executing temporal logic programs. In *International Conference on Concurrency*, pages 111–130. Springer, 1984.

[21] Amir Pnueli. The temporal logic of programs. In *Foundations of Computer Science, 1977., 18th Annual Symposium on*, pages 46–57. IEEE, 1977.

[22] Arthur N. Prior. *Time and Modality.* Clarendon Press Oxford, 1957.

[23] Arthur N. Prior. *Past, present and future.* Clarendon Press Oxford, 1967.

[24] Anders P Ravn. *Design of embedded real-time computing systems.* PhD thesis, Technical University of Denmark Lyngby, 1995.

[25] Roni Rosner and Amir Pnueli. *A choppy logic.* Weizmann Institute of Science. Department of Applied Mathematics, 1986.

[26] Dogan Ulus. Montre: A tool for monitoring timed regular expressions. In *Computer Aided Verification (CAV)*, 2017.

[27] Dogan Ulus, Thomas Ferrère, Eugene Asarin, and Oded Maler. Timed pattern matching. In *Formal Modeling and Analysis of Timed Systems (FORMATS)*, pages 222–236, 2014.

[28] Dogan Ulus, Thomas Ferrère, Eugene Asarin, and Oded Maler. Online timed pattern matching using derivatives. In *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, pages 736–751, 2016.

[29] Yde Venema. Expressiveness and completeness of an interval tense logic. *Notre Dame Journal of Formal Logic*, 31(4):529–547, 1990.

[30] Yde Venema. A modal logic for chopping intervals. *Journal of Logic and Computation*, 1(4):453–476, 1991.

[31] Masaki Waga, Takumi Akazaki, and Ichiro Hasuo. A boyer-moore type algorithm for timed pattern matching. In *International Conference on Formal Modeling and Analysis of Timed Systems*, pages 121–139. Springer, 2016.

[32] Masaki Waga, Ichiro Hasuo, and Kohei Suenaga. Efficient online timed pattern matching by automata-based skipping. In *International Conference on Formal Modeling and Analysis of Timed Systems*, pages 224–243. Springer, 2017.