Pattern Matching and Parameter Identification for Parametric Timed Regular Expressions

Akshay Mambakam, Eugene Asarin, Nicolas Basset, Thao Dang

Pattern Matching and Parameter Identification for Parametric Timed Regular Expressions

Akshay Mambakam
Akshay.Mambakam@univ-grenoble-alpes.fr
Université Grenoble Alpes, CNRS, Grenoble INP, VERIMAG
Grenoble, France

Eugene Asarin
asarin@irif.fr
Université Paris Cité, CNRS, IRIF
Paris, France

Nicolas Basset
nicolas.basset1@univ-grenoble-alpes.fr
Université Grenoble Alpes, CNRS, Grenoble INP, VERIMAG
Grenoble, France

Thao Dang
thao.dang@univ-grenoble-alpes.fr
Université Grenoble Alpes, CNRS, Grenoble INP, VERIMAG
Grenoble, France

ABSTRACT

Timed formalisms such as Timed Automata (TA), Signal Temporal Logic (STL) and Timed Regular expressions (TRE) have been previously applied as behaviour specifications for monitoring or runtime verification, in particular, under the form of pattern-matching, *i.e.* computing the set of all the segments of a given system run that satisfy the specification.

In this work, timed regular expressions with parameters (for timing delays and for signal values) are considered. We define several classes of parametric expressions (based on Boolean or real-valued signals and discrete events), and tackle the problem of computing a parametric match-set, *i.e.* the parameter values and time segments of data that give a match for a given expression. We propose efficient data structures for representing match-sets (combining zones and polytopes), and devise pattern-matching algorithms. All these different types and algorithms are combined into a single implementation under a tool named `parameTRE`. We illustrate the approach on several examples, from electrocardiograms to driving patterns.

CCS CONCEPTS

• **Theory of computation → Timed and hybrid models**; • **Computing methodologies → Algebraic algorithms**.

KEYWORDS

Timed Languages, Pattern Matching, Monitoring, Parametric Identification

ACM Reference Format:
Akshay Mambakam, Eugene Asarin, Nicolas Basset, and Thao Dang. 2023. Pattern Matching and Parameter Identification for Parametric Timed Regular Expressions. In *Proceedings of the 26th ACM International Conference on Hybrid Systems: Computation and Control (HSCC 2023), May 9–12, 2023, San Antonio, TX, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3575870.3587115

HSCC 2023, May 9–12, 2023, San Antonio, TX, USA

ACM ISBN 979-8-4007-0033-0/23/05…$15.00
https://doi.org/10.1145/3575870.3587115

1 INTRODUCTION

Cyber-physical systems involving dynamic interaction between computing devices and physical environments, often critical, have become omnipresent in the modern world. Formal verification thereof, *i.e.* checking specified properties over all possible behaviors of a system, is highly desirable but limited by decidability and complexity issues. This explains a growing interest in monitoring, runtime verification and other light-weight forms of formal verification. These approaches involve checking properties of individual behaviours. Also, monitoring properties of signals (or time series) is relevant in other domains, such as to detect arrhythmia events in an electrocardiogram (ECG) or a vehicle collision in road traffic. Using a formal specification language one can automatically generate property monitors. This can replace the tedious task of manually inspecting or writing ad-hoc property monitors.

Several formalisms have been used to specify system behaviours on the timed level, among which we can mention Timed Automata (TA) [1], Timed Regular Expressions (TRE) [4], Metric Interval Temporal Logic (MITL) [19]. On the level of real-valued and Boolean signals, Signal Temporal Logic (STL) [14, 18] is often used. For such formalisms, monitoring problems have been often phrased as pattern-matching: given a behaviour and a specification (a pattern), detect all the time intervals when the pattern occurs. Efficient algorithms have been developed for matching patterns specified by TA, or various kinds of TRE and temporal logics [8, 11, 15, 21, 22].

A further step in this research direction is introducing parameters in the specifications, which has numerous advantages:

– Parametric formalisms are more expressive and flexible. Without parameters, delays and thresholds in a specification should be constant. It is possible to specify that within 23 seconds after braking the motor stops, or that whenever after 30 minutes of overheating (that happens if $T > 130$) a pipe explodes (event E). With parameters one can describe a generic pattern: after θ minutes of $(T > \sigma)$ the event E occurs. Thus one can describe event occurrences of signals of particular form but with a-priory unknown scale.

- One can learn typical values of parameters that allow pattern matching with observed data. This problem is solved in the framework of parametric pattern matching (see [6]).
- Parametric pattern matching can be used for quantitative runtime verification: for example characteristics such as maximal or minimum response times can be specified as matching parameters.
- In addition, graphs of matching parameters would provide engineers or doctors with precious visual information on system behaviours, which is easier to interpret and requires less training than manual ad-hoc inspection.

Parametric specifications have already been considered in the context of timed systems. First, parametric timed automata (PTA) have been introduced in [2], and undecidability of empty language and other model-checking problems has been established. Years later, [6, 9] considered identification of timing and magnitude parameters in a parametric version of STL logic called Parametric Signal Temporal Logic (PSTL), in the context of runtime verification. Recently [3, 23–25], with a motivation similar to ours, considered parametric timed pattern matching with respect to parametric timed automata.

The main contribution of this article is the development and implementation of algorithms of parametric timed pattern-matching for several kinds of parametric variants of Timed Regular Expressions. Before continuing with technical formalisation, let us illustrate parametric expressions by a simple example which concerns two Boolean signals p and q depicted on Fig. 1. We consider the

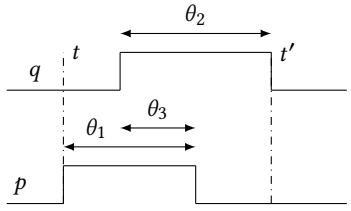


Figure 1: Matching a Parametric Timed Regular Expression ϕ_1

following expression:

$$\phi_1 := (\langle p \rangle_{\theta_1} \cdot \neg p) \wedge (\neg q \cdot \langle q \rangle_{\theta_2}) \wedge (\mathtt{true} \cdot \langle p \wedge q \rangle_{\theta_3} \cdot \mathtt{true}) \quad (1)$$

It represents the intersection of three different expressions. The first expression represents a signal p of duration θ_1. The second one represents a signal q of duration θ_2. Finally, the third expression represents the conjunction of p and q with a duration of θ_3. The period from t to t' matches the parametric expressions. We are interested in finding the *match-set*, that is the set of all the tuples $(t, t', \theta_1, \theta_2, \theta_3)$ corresponding to such matches.

Such match-sets can be computed using the parametric timed pattern matching algorithms we propose in this paper. More concretely, the algorithms take as input a timed Boolean or real-valued signal (or event sequence), and a parametric specification. The algorithms output a set of matching time intervals together with parameter values (represented by a special data structure). Our algorithms proceed by structural induction over the regular expression, inspired by [21], and combine two set representation technologies for match-sets: one based on zones (a well-known data structure in the domain of timed automata and suitable for representing intervals of time), and another using polytopes for parameter values.

This paper is structured as follows. In Section 2 we define several kinds of timed behaviours, and recall Timed Regular Expressions (TRE). In Section 3 we describe several flavors of parametric expressions, and for each of them characterize the match-set. In Section 4 we develop our practical pattern-matching algorithms. In Section 5 we deal with the problem of parametric identification and in Section 6 we present our experimental results, followed by a comparison with previous work in Section 7. We conclude and outline some perspectives on future work in Section 8.

2 PRELIMINARIES

In this section, we give some basic definitions and explain the syntax and semantics of (non-parametric) TRE [21]. We assume a time domain $\mathbb{T} = [0, d]$ which is a bounded interval of $\mathbb{R}_+$. Let n and m be positive integers. In the rest of the article we consider a set of real variables $X = \{x_1, x_2, ..., x_n\}$ and a set of propositional variables $P = \{p_1, p_2, ..., p_m\}$.

Definition 1 (Signals). A signal is a function $w : \mathbb{T} \to \mathbb{R}^n \times \mathbb{B}^m$. At each time point in $\mathbb{T}$ the signal assigns real values to variables $x \in X$ and Boolean values to variables $p \in P$. They are called real-valued and Boolean if they are of the form $x : \mathbb{T} \to \mathbb{R}^n$ and $x : \mathbb{T} \to \mathbb{B}^m$ respectively.

The value of x at time t is denoted by $x[t]$ and its i^{th} coordinate by $x_i[t]$. Abusing the notation, we often use the variable names $x_1, \ldots, x_n, p_1, \ldots, p_m$ both for the values of the signal and in the expression.

Definition 2 (Finite Variability). Let $w : \mathbb{T} \to \mathbb{R}^n \times \mathbb{B}^m$ be a signal. Let us also assume that we are given a set of atomic predicates Π of the form $(x \geq c)$ or $(x \leq c)$ where $x \in X$ and $c \in \mathbb{R}$. Each of these predicates in Π produces a Boolean signal $t \mapsto x(t) \geq c$. The signal w is said to be of finite variability if for every propositional variable/atomic predicate in $P \cup \Pi$ the corresponding Boolean signal has a finite set of discontinuities (See [15] for details).

From here on, the Boolean signals we talk about are assumed to be of finite variability. Also, the real-valued signals considered are assumed to be piecewise affine so that the Boolean signals they produce using atomic predicates (like $x \geq c$) are also of finite variability.

Definition 3 (Timed Words). A timed word of length l over an alphabet Σ is a sequence $\omega = t_1 a_1 ... t_l a_l$ with $a_i \in \Sigma, t_i \in \mathbb{R}$ and $0 \leq t_1 \leq ... \leq t_l$. Here, t_i represents the date at which the event a_i occurs.

The following definition concerns Boolean signals, that is, with a non-empty set of Boolean variables P and an empty set of real-valued variables X.

Definition 4 (Timed Regular Expressions (TRE)). The syntax of Timed Regular Expressions is given by the grammar

$$\varphi := \epsilon \mid p \mid \overline{p} \mid \langle \phi \rangle_I \mid \phi \cdot \psi \mid \phi^* \mid \phi \wedge \psi \mid \phi \vee \psi$$

where $p \in P$ is a propositional variable and I is an interval of $\mathbb{R}_+$ with integer endpoints.

Unlike Signal Temporal Logic where satisfaction of a formula is of the form $(w,t) \models \varphi$, for TRE we define satisfaction in terms of two time points t, t'. This means that $(w,t,t') \models \varphi$ represents the fact that $w[t,t']$ (the factor of w defined over the sub-domain $[t,t']$) satisfies the semantics of the expression φ.

Definition 5 (TRE Semantics). The satisfaction relation $\models$ of a TRE φ by a signal w, relative to start time t and end time $t' \geq t$ is defined as follows:

$$
\begin{aligned}
&(w,t,t') \models \epsilon \leftrightarrow t = t' \\
&(w,t,t') \models p \leftrightarrow t < t' \wedge \forall t'' \; t < t'' < t' \rightarrow p[t''] = 1 \\
&(w,t,t') \models \overline{p} \leftrightarrow t < t' \wedge \forall t'' \; t < t'' < t' \rightarrow p[t''] = 0 \\
&(w,t,t') \models \varphi \cdot \psi \leftrightarrow \exists t'' \cdot (w,t,t'') \models \varphi \wedge (w,t'',t') \models \psi \\
&(w,t,t') \models \varphi \vee \psi \leftrightarrow (w,t,t') \models \varphi \vee (w,t,t') \models \psi \\
&(w,t,t') \models \varphi \wedge \psi \leftrightarrow (w,t,t') \models \varphi \wedge (w,t,t') \models \psi \\
&(w,t,t') \models \varphi^* \leftrightarrow \exists k \geq 0 \cdot (w,t,t') \models \varphi^k \\
&(w,t,t') \models \langle \varphi \rangle_I \leftrightarrow t' - t \in I \wedge (w,t,t') \models \varphi
\end{aligned}
$$

Definition 6 (Match-Set). For any signal w and expression φ, we define the match-set as:

$$\mathcal{M}(\varphi, w) := \{(t,t') \in \mathbb{T} \times \mathbb{T} : (w,t,t') \models \varphi\}$$

Geometrically, match-sets are subsets of the upper triangular portion defined by $t \leq t'$, of the box $[0,d] \times [0,d]$. In [21] it is shown that for every TRE φ and a finite-variability signal w, the match-set can be written as a finite union of zones. A zone is of the form $(t \prec c_1) \wedge (t' \prec c_2) \wedge (t' - t \prec c_3) \wedge (c_4 \prec t' - t) \wedge (c_5 \prec t') \wedge (c_6 \prec t)$ where $c_1, c_2, \ldots, c_6$ are constants and the symbol $\prec$ is $\leq$ / $<$.

Introducing atomic constraints such as $x \leq c$ and $x \geq c$ (where c is a constant) in TRE leads to (non-parametric) Signal Regular Expressions (SRE) [10]. We will introduce parametric SRE in Subsection 3.1.

3 PARAMETRIC TIMED REGULAR EXPRESSIONS

In this section, we introduce parameters into three different classes of TRE and derive properties of their parametric match-sets. The first resulting class is a parametric version of Signal Regular Expressions (SRE) [10] which is apt for expressing behaviours of real-valued signals. The second is a parametric version of (event-based) Timed Regular Expressions [5] suited for time-event sequences. The third is a parametric version of event-bounded TRE [15] which is a hybrid formalism mixing state-based and event-based features, with events captured as rising or falling edges of Boolean signals, combined with SRE. We also introduce the notions of parametric zones and parametric intervals and show how parametric match-sets can be expressed using these set representations.

3.1 Parametric Signal Regular Expressions (PSRE)

We are given a signal $w : \mathbb{T} \rightarrow \mathbb{R}^n \times \mathbb{B}^m$ over a set of real variables $X = \{x_1, x_2, ..., x_n\}$ and a set of propositional variables $P = \{p_1, p_2, ..., p_m\}$. At each time point in $\mathbb{T}$ the signal assigns real values to variables $x \in X$ and Boolean values to variables $p \in P$. In PSRE, there are two types of parameters, *magnitude* parameters denoted by a vector $q = (q_1, ..., q_g)$ and *timing* parameters denoted by a vector $s = (s_1, ..., s_h)$. Their domains are polytopes $\mathcal{Q} \subseteq \mathbb{R}^g$ and $\mathcal{S} \subseteq [0,\infty)^h$. A syntactic description of PSRE is as follows:

Definition 7 (Parametric Signal Regular Expressions).

$$x \geq \lambda \mid x \leq \lambda \mid p \mid \overline{p} \mid \langle \phi \rangle_I \mid \phi \cdot \psi \mid \phi^* \mid \phi \wedge \psi \mid \phi \vee \psi$$

where $x \in X$ is a real-valued variable, λ is either a constant or a magnitude parameter q_i, $p \in P$ a propositonal variable, and I stands for an interval $[a,b]$ where each of a, b is either a non-negative constant (timing constant) or a timing parameter s_i.

Here x corresponds to a real-valued signal, and p a Boolean one. A parametric valuation $(u,v) \in \mathcal{Q} \times \mathcal{S}$ converts a PSRE formula φ into a SRE formula $\varphi_{u,v}$ obtained by substituting the values (u,v) in the parameters (q,s). We use the notations $\lambda_{u,v}$ and $I_{u,v}$ to denote the threshold and interval respectively obtained from such a substitution. The semantics of a PSRE φ with respect to a signal w is given in terms of a parametric match set.

Definition 8 (PSRE Semantics). The satisfaction relation $\models$ of a PSRE φ by a signal w with respect to a start time t, a end time t' and a parametric valuation (u,v) is defined inductively as follows:

$$
\begin{aligned}
&(w,t,t',u,v) \models (x \leq \lambda) \leftrightarrow t < t' \wedge \forall t''.t < t'' < t' \rightarrow x[t''] \leq \lambda_{u,v} \\
&(w,t,t',u,v) \models (x \geq \lambda) \leftrightarrow t < t' \wedge \forall t''.t < t'' < t' \rightarrow x[t''] \geq \lambda_{u,v} \\
&(w,t,t',u,v) \models p \leftrightarrow t < t' \wedge \forall t''.t < t'' < t' \rightarrow p[t''] = 1 \\
&(w,t,t',u,v) \models \overline{p} \leftrightarrow t < t' \wedge \forall t''.t < t'' < t' \rightarrow p[t''] = 0 \\
&(w,t,t',u,v) \models \langle \varphi \rangle_I \leftrightarrow t' - t \in I_{u,v} \wedge (w,t,t',u,v) \models \varphi \\
&(w,t,t',u,v) \models \varphi \cdot \psi \leftrightarrow \exists t''.(w,t,t'',u,v) \models \varphi \wedge (w,t'',t',u,v) \models \psi \\
&(w,t,t',u,v) \models \varphi \wedge \psi \leftrightarrow (w,t,t',u,v) \models \varphi \text{ and} \\
&(w,t,t',u,v) \models \psi \\
&(w,t,t',u,v) \models \varphi \vee \psi \leftrightarrow (w,t,t',u,v) \models \varphi \text{ or} \\
&(w,t,t',u,v) \models \psi \\
&(w,t,t',u,v) \models \varphi^* \leftrightarrow \exists k \geq 0.(w,t,t',u,v) \models \varphi^k
\end{aligned}
$$

Definition 9 (Parametric Match-Set). For any signal w and PSRE φ, we define the parametric match-set

$$\mathcal{M}(\varphi, w) = \{(t,t',u,v) \in \mathbb{T} \times \mathbb{T} \times \mathcal{Q} \times \mathcal{S} : (w,t,t',u,v) \models \varphi\}$$

3.1.1 PSRE Example. Let us consider the following expression which can approximately match an electrocardiogram signal.

$$\phi_2 := \langle -0.55 \leq x \leq 0.29 \rangle_{\theta_1} \cdot \langle 0.29 \leq x \leq 2.0 \rangle_{\theta_2} \cdot \langle -0.6 \leq x \leq 0.29 \rangle_{\theta_3} \quad (2)$$

In ϕ_2 (see Expr (2)), the signal x stays in the interval $[-0.55, 0.29]$ for a duration of θ_1 time units. Then, its value increases and stays within $[0.29, 2.0]$ for θ_2 time units. Finally, it decreases and stays within $[-0.6, 0.29]$ for θ_3 time units. Figure 2 illustrates a matching signal.

3.1.2 Parametric Zones and Parametric Match-Sets. Recall that we represent the vectors of magnitude and timing parameters with the symbols q and s respectively. A parametric zone is defined by six constraints of the following form:

$$\mathcal{T}(c_1,c_2,c_3,c_4,c_5,c_6) := (t \prec c_1(q,s)) \wedge (t' \prec c_2(q,s)) \wedge (t' - t \prec c_3(q,s)) \wedge (c_4(q,s) \prec t' - t) \wedge (c_5(q,s) \prec t') \wedge (c_6(q,s) \prec t)$$

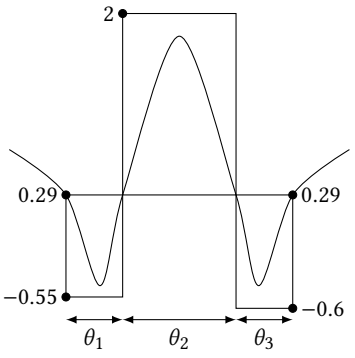


Figure 2: ECG matching expression ϕ_2

along with a set of linear constraints $\boldsymbol{C_z}(q,s)$ over the parameter space, where $c_1, ..., c_6$ are piece-wise linear functions over (q,s). More precisely, c_1, c_2, c_3 can be expressed as minima of linear functions and c_4, c_5, c_6 as maxima of linear functions. The symbol $\prec$ is $\leq / <$.

Theorem 10. *For signals that are Boolean or piecewise linear, the parametric match-set of PSRE is a finite union of parametric zones.*

Proof. Let us consider the operators one by one and show that they preserve the form when applied over parametric zones.

From the proofs in [21], it can be inferred that the parametric match sets $\mathcal{M}(p,w)$ and $\mathcal{M}(\overline{p},w)$ for Boolean signal p, is a set of triangular cylinders touching the diagonal $t \leq t'$.

The parametric match set for $(x \geq \lambda)$ is

$$\mathcal{M}(x \geq \lambda, w) = \{(t,t',\lambda) \mid \lambda \leq \inf_{t'' \in (t,t')} w_x[t'']\},$$

which is equivalent to the robustness support [10] for $(x \geq 0)$, that is

$$\mathcal{R}(x \geq 0, w) = \{(t,t',r) \mid r \leq \inf_{t'' \in (t,t')} w_x[t'']\}.$$

Similarly $\mathcal{R}(x \leq 0, w)$ is equivalent to $\mathcal{M}(x \leq \lambda, w)$. In [10], it is shown that the robustness support of $x \leq 0$ and $x \geq 0$ for piecewise linear and piece-wise constant signals can be represented as a finite union of polytopes (which are indeed parametric zones). Therefore, it follows that the parametric match-set for $x \geq \lambda$ and $x \leq \lambda$ is a finite union of parametric zones for these classes of signals. Now, for the disjunction operator, we have $\mathcal{M}(\varphi \vee \psi, w) = \mathcal{M}(\varphi, w) \cup \mathcal{M}(\psi, w)$. Note that a finite union of parametric zones is closed under the disjunction operator.

Next, the parametric match set for conjunction is $\mathcal{M}(\varphi \wedge \psi, w) = \mathcal{M}(\varphi, w) \cap \mathcal{M}(\psi, w)$.

Let us consider two parametric zones: $\mathbf{z}_1 = \mathcal{T}(c_1, c_2, c_3, c_4, c_5, c_6) \wedge \boldsymbol{C_{z_1}}$ and $\mathbf{z}_2 = \mathcal{T}(c'_1, c'_2, c'_3, c'_4, c'_5, c'_6) \wedge \boldsymbol{C_{z_2}}$.

We have, $\mathbf{z}_1 \cap \mathbf{z}_2 = \mathcal{T}(\min(c_1, c'_1), \min(c_2, c'_2), \min(c_3, c'_3), \max(c_4, c'_4), \max(c_5, c'_5), \max(c_6, c'_6)) \wedge \boldsymbol{C_{z_1}} \wedge \boldsymbol{C_{z_2}}$. Finite union of parametric zones is also closed under the conjunction operator.

For the concatenation operator, the parametric match set for concatenation is $\mathcal{M}(\varphi \cdot \psi, w) = \mathcal{M}(\varphi, w) \circ \mathcal{M}(\psi, w)$. The symbol $\circ$ is called sequential composition which is an operation over zones defined as $\mathbf{z}_1 \circ \mathbf{z}_2 = \mathcal{T}(\min(c_1, c'_1 - c_4, c_2 - c_4), \min(c'_2, c_2 + c'_3, c'_1 + c'_3), c_3 + c'_3, c_4 + c'_4, \max(c'_5, c_5 + c'_4, c'_6 + c'_4), \max(c_6, c'_6 - c_3, c_5 - c_3)) \wedge \boldsymbol{C_{z_1}} \wedge \boldsymbol{C_{z_2}} \wedge (c_5 \prec c'_1) \wedge (c'_6 \prec c_2) \wedge (c_4 \prec c_3) \wedge (c'_4 \prec c'_3) \wedge (c_5 \prec c_2) \wedge (c'_6 \prec c'_1)$ The above equation for sequential composition is obtained by performing Fourier-Motzkin quantifier elimination. It follows that, a finite union of parametric zones is closed under concatenation operator.

For duration restriction, the parametric match set is $\mathcal{M}(\langle\varphi\rangle_{[a,b]}, w) = \mathcal{M}(\varphi, w) \cap \{(t,t') : t' - t \in [a,b]\}$.

For a zone $\mathbf{z}_1 = \mathcal{T}(c_1, c_2, c_3, c_4, c_5, c_6) \wedge \boldsymbol{C_{z_1}}$, we have:

$\mathbf{z}_1 \wedge \{(t,t') : t' - t \in [a,b]\} = \mathcal{T}(c_1, c_2, \min(c_3, b), \max(c_4, a), c_5, c_6) \wedge \boldsymbol{C_{z_1}}$

Again, we can see that a finite union of parametric zones is closed under duration restriction. □

3.1.3 Kleene Star and Finite Number of Concatenations. In this subsection we prove that the parametric match-set of Kleene star of PSRE can be computed by a finite number of concatenations under a mild hypothesis on the set of parameters.

We start by recalling some definitions followed by Lemma 11 from [21] established for the non-parametric case. An interval $[t,t']$ is said to be *unitary* with respect to a signal w if $t' - t < 1$ and w is constant throughout its interior (t,t'). Let $\sigma(w)$ be the least k such that w can be covered by k unitary intervals. A key property of $m = \sigma(w)$ is stated in the following lemma.

Lemma 11. *[21]. For any $n > 2m+1$ if $(w,t,t') \models \varphi^n$ then $(w,t,t') \models \varphi^{n-1}$ for an SRE φ.*

We define w_u as the set of Boolean signals obtained by introducing a magnitude parametric valuation u in a PSRE φ. Let $m' = 2 + 2 \max_{q \in Q} \sigma(w_q)$.

Lemma 12. *Kleene star can be bounded as follows: $(w,t,t',u,v) \models \varphi^*$ if and only if $(w,t,t',u,v) \models \varphi^n$ for some $n \leq m'$.*

Proof. It is easy to see that for piecewise constant and piecewise linear signals, m indeed has finite value. For the sake of contradiction, let us assume the following,

$$\exists (n > m'), t, t', u, v \cdot (w,t,t',u,v) \models \varphi^n \wedge (w,t,t',u,v) \not\models \varphi^{(n-1)}.$$

Simplifying this we get, $(w_{u_0}, t_0, t'_0) \models \varphi_{u_0,v_0}^{n_0} \wedge (w_{u_0}, t_0, t'_0) \not\models \varphi_{u_0,v_0}^{n_0-1}$ where n_0, t_0, t'_0, u_0 and v_0 are constants with $n_0 > m'$. From the definition of m' we can deduce that $n_0 > 2\sigma(w_{u_0}) + 2$. This contradicts Lemma 11 for the case of signal w_{u_0} and expression φ_{u_0,v_0}. □

3.2 Parametric Timed Regular Expressions with Event-Based Semantics

In this subsection, we consider Parametric Timed Regular Expressions (PTRE) with event-based semantics that apply to timed words rather than to signals. The vector of timing parameters $(s_1, ..., s_h)$ is again denoted by s. Note that this version of PTRE does not have magnitude parameters.

Definition 13 (Parametric Timed Regular Expressions with Event-Based Semantics).

$$\underline{a} \mid \epsilon \mid \langle\varphi\rangle_I \mid \varphi_1 \cdot \varphi_2 \mid \varphi_1 \cup \varphi_2 \mid \varphi_1 \cap \varphi_2 \mid \varphi^*$$

where $I = [\alpha, \beta]$ and each of α, β is either a non-negative constant or a timing parameter s_i. For all $a \in \Sigma$ we define $\underline{a}$ which represents

an arbitrary passage of time followed by event a where Σ is the event alphabet over which the PTRE is defined.

The domain of parameters $\mathcal{S} \subseteq [0,\infty)^h$ is expressed using a polytope. A parametric valuation $v \in \mathcal{S}$ converts a PTRE formula φ into a TRE formula φ_v obtained by substituting the values v in parameters s. We use I_v to denote the interval obtained from such a substitution.

Definition 14 (Parametric Match-Set). The parametric match-set of a PTRE expression φ (with event-based semantics) for a timed word $\omega = t_1 a_1 ... t_n a_n$ is defined inductively as follows ($t_0 = 0$ by default):

$$\mathcal{M}(\underline{a}, \omega) := \{(t, t', v) : \exists i \in [1...n] \cdot a = a_i \wedge t = t_{i-1} \wedge t' = t_i\}$$
$$\mathcal{M}(\epsilon, \omega) := \{(t, t, v) : \exists i \in [1...n] \cdot t = t_{i-1} \wedge t = t_i\}$$
$$\mathcal{M}(\langle\varphi\rangle_I, \omega) := \{(t, t', v) : t' - t \in I_v \wedge (t, t', v) \in \mathcal{M}(\varphi, \omega)\}$$
$$\mathcal{M}(\varphi \cdot \psi, \omega) := \{(t, t', v) : \exists t''.(t, t'', v) \in \mathcal{M}(\varphi, \omega) \wedge (t'', t', v) \in \mathcal{M}(\psi, \omega)\}$$
$$\mathcal{M}(\varphi \wedge \psi, \omega) := \{(t, t', v) : (t, t', v) \in \mathcal{M}(\varphi, \omega) \wedge (t, t', v) \in \mathcal{M}(\psi, \omega)\}$$
$$\mathcal{M}(\varphi \vee \psi, \omega) := \{(t, t', v) : (t, t', v) \in \mathcal{M}(\varphi, \omega) \vee (t, t', v) \in \mathcal{M}(\psi, \omega)\}$$
$$\mathcal{M}(\varphi^*, \omega) := \{\exists k \geq 0.(t, t', v) \in \mathcal{M}(\varphi^k, \omega)\}$$

3.2.1 Parametric Intervals and Parametric Match-Sets. A parametric interval is defined by constraints of form $t = c_1$, $t' = c_2$ and $\boldsymbol{C_y}(r)$ which is a set of linear constraints over the parameter set r. Given constants c_1 and c_2, the constraints $t = c_1$ and $t' = c_2$ represent the beginning and ending of a time interval. Theorem 15 shows that the parametric match-sets of PTRE can be expressed as unions of parametric intervals and its proof provides a constructive procedure for match-set computation.

Theorem 15. *For a timed word the parametric match-set of a PTRE is a finite union of parametric intervals.*

Proofs of the above Theorem 15 and that of Theorem 16 (in Section 3.3) are given in Appendix B.

3.2.2 PTRE Example. Let us consider the following PTRE which has both concatenation and intersection operators:

$$\phi_3 := (\langle \underline{a} \cdot \underline{b} \rangle_{\theta_1} \cdot \underline{c}) \wedge (\underline{a} \cdot \langle \underline{b} \cdot \underline{c} \rangle_{\theta_2}) \tag{3}$$

Recall that $\underline{a}$ represents an arbitrary passage of time followed by the event a. This can be represented as $r \cdot a$, where r represents passage of time and a is an event. Therefore we can write the semantics of $\underline{a}$ as, $[[\underline{a}]] := \{r \cdot a : r \in \mathbb{R}_+\}$.

The semantics of ϕ_3 (Expr (3)) containing parameters θ_1 and θ_2 can be deduced as follows: $\{r_1 \cdot a \cdot r_2 \cdot b \cdot r_3 \cdot c : (r_1 + r_2 = \theta_1) \wedge (r_2 + r_3 = \theta_2)\}$.

3.3 Parametric Event-Bounded Timed Regular Expressions

We now consider another parametric extension of TRE called event-bounded TRE (E-TRE). We are given a signal w defined exactly the same as in Section 3.1. The signal w assigns to each propositional variable $p \in P$ a Boolean value at each time point t in the time domain $\mathbb{T}$. So, for each $p \in P$ we have a corresponding Boolean signal. A parametric E-TRE (PE-TRE) is of form $\uparrow p$, $\psi_1 \cdot \varphi \cdot \psi_2$, $\psi_1 \vee \psi_2$, or $\psi_1 \wedge \varphi$ where $p \in P$, $\uparrow p$ stands for a rising edge of p, ψ_1 and ψ_2 are PE-TRE and φ stands for a PSRE. Note that falling edge can be defined as $\downarrow p := \uparrow \overline{p}$.

Theorem 16. *For Boolean, piecewise linear and piecewise constant signals the parametric match-set of PE-TRE is always a finite union of parametric intervals.*

3.3.1 PE-TRE Example. ϕ_4 (Expression (4)) denotes the pattern of a brake control signal b for a vehicle under heavy braking situation. It is a parameterized version of the brake control signal pattern of the anti-lock brake system example given in [15]. It starts with a rise edge of b followed by a braking period of duration less than θ_1. It continues with one or more pulses with duration less than θ_2. It ends with a falling edge of b. In Figure 3, we can see an illustration of the braking pattern b.

$$\phi_4 := \uparrow b \cdot \langle b \rangle_{[0,\theta_1]} \cdot \langle \neg b \cdot b \rangle^+_{[0,\theta_2]} \cdot \downarrow b \tag{4}$$

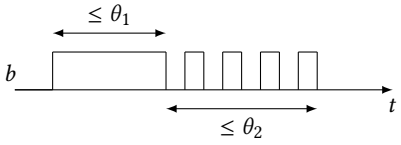


Figure 3: Braking Pattern

4 PARAMETRIC MATCH-SET COMPUTATION

In this section, we discuss how to compute parametric match-sets. As stated in Theorems 10, 15, 16, the match-sets can be represented using unions of parametric zones or parametric intervals. The abstract procedures for their construction is given in the proofs of these theorems. In this section we focus on concrete computational algorithms for performing the required operations: intersection and concatenation in Subsections 4.1, transitive closure in Subsection 4.3.

4.1 Binary Operations of PSRE and PE-TRE

We remark that some sub-expressions of PE-TRE have parametric match-sets that are unions of parametric zones, and all sub-expressions of PSRE have parametric match-sets that are unions of parametric zones. Therefore, for both PE-TRE and PSRE the parametric match-sets for sub-expressions are represented as unions of polytopes. We inductively perform binary zone operations (*intersection* and *concatenation*) over unions of polytopes. The resulting final match-set for the whole expression will also be a union of polytopes. A naive implementation of intersection and concatenation operations would involve $O(n^2)$ of polytope intersections and sequential compositions respectively. However, if we exploit the inherent temporal ordering between parametric zones, this can be avoided. Indeed, using the plane-sweep idea, we need only to consider pairs of polytopes which potentially overlap. Note that this idea was also used in [21].

We first define the functions $\pi_1^+, \pi_1^-, \pi_2^+, \pi_2^-$ over parametric zones as below:

$$\pi_1^+(\mathbf{z}) = \{\max(t) : (t, t', u, v) \in \mathbf{z}\}$$

$$\pi_1^-(\mathbf{z}) = \{\min(t) : (t, t', u, v) \in \mathbf{z}\}$$

$$\pi_2^+(\mathbf{z}) = \{\max(t') : (t, t', u, v) \in \mathbf{z}\}$$

$$\pi_2^-(\mathbf{z}) = \{\min(t') : (t, t', u, v) \in \mathbf{z}\}$$

They can be computed using convex optimization over parametric zones (polytopes). With the values of these functions we can bound each zone inside a rectangular cylinder over the timed dimensions t, t'. For intersection, Algorithm 1, we sort the lists Z and Z' according to π_1^-. We keep two active lists Y and Y' that contain the candidates for intersection. We move elements one by one to the active lists and remove them when we deduce that they will not participate in further non-empty intersections. We can remove a $z \in Y$ if $\pi_1^+(z) < \pi_1^-(z')$ for every $z' \in Y'$ and vice versa. For concatenation, Algorithm 2, we sort Z by π_2^- and Z' by π_1^-. We then compute all pairs $z \circ z'$ such that $[\pi_2^-(z), \pi_2^+(z)] \cap [\pi_1^-(z'), \pi_1^+(z')] \neq \emptyset$. This means that we discard the combinations where the end interval for $z \in Z$ does not intersect the begin interval for $z' \in Z'$. Note that first($\mathbf{Z}$) denotes the first element of the ordered list $\mathbf{Z}$.

Algorithm 1 INTERSECT(Z, Z')
assume Z, Z' sorted by π_1^-

for each $z \in Z$ **do**
  Compute $\pi_1^+, \pi_1^-, \pi_2^+, \pi_2^-$
for each $z' \in Z'$ **do**
  Compute $\pi_1^+, \pi_1^-, \pi_2^+, \pi_2^-$
$Y := Y' := Z'' := \emptyset$
while $Z \neq \emptyset \vee Z' \neq \emptyset$ **do**
  $z := \text{first}(Z); l := \pi_1^-(z)$
  $z' := \text{first}(Z'); l' := \pi_1^-(z')$
  if $l < l'$ **then**
    Move z from Z to Y
    $Y' = \{z' \in Y' : \pi_1^+(z') \geq l\}$
    for each $z' \in Y'$ **do**
      $z'' := z \cap z'$ ▷ Polyhedra intersection
      $Z'' := Z'' \cup z''$ ▷ Check emptiness before adding
  else
    Move z' from Z' to Y'
    $Y = \{z \in Y : \pi_1^+(z) \geq l'\}$
    for each $z \in Y$ **do**
      $z'' := z \cap z'$
      $Z'' := Z'' \cup z''$ ▷ Check emptiness before adding
return Z''

4.2 Binary Operations in PTRE (with Event-Based Semantics)

All sub-expressions of PTRE (Event-Based) have parametric match sets that are union of parametric intervals. To efficiently perform the intersection and concatenation operations over unions of parametric intervals, sorting and binary search algorithms can be used. Given a parametric interval $\mathbf{y} := (t = d_1 \wedge t' = d_2 \wedge \boldsymbol{C_y})$ we define the functions π^- and π^+ as $\pi^-(\mathbf{y}) = d_1$ and $\pi^+(\mathbf{y}) = d_2$ respectively.

Algorithm 2 CONCAT(Z, Z')
assume Z sorted by π_2^-, Z' sorted by π_1^-

for each $z \in Z$ **do**
  Compute $\pi_1^+, \pi_1^-, \pi_2^+, \pi_2^-$
for each $z' \in Z'$ **do**
  Compute $\pi_1^+, \pi_1^-, \pi_2^+, \pi_2^-$
$Y := Y' := Z'' := \emptyset$
while $Z \neq \emptyset \vee Z' \neq \emptyset$ **do**
  $z := \text{first}(Z); l := \pi_2^-(z)$
  $z' := \text{first}(Z'); l' := \pi_1^-(z')$
  if $l < l'$ **then**
    Move z from Z to Y
    $Y' = \{z' \in Y' : \pi_1^+(z') \geq l\}$
    for each $z' \in Y'$ **do**
      $z'' := z \circ z'$ ▷ Polyhedra renaming, intersection and quantifier elimination
      $Z'' := Z'' \cup z''$ ▷ Check emptiness before adding
  else
    Move z' from Z' to Y'
    $Y = \{z \in Y : \pi_2^+(z) \geq l'\}$
    for each $z \in Y$ **do**
      $z'' := z \circ z'$
      $Z'' := Z'' \cup z''$ ▷ Check emptiness before adding
return Z''

For intersection, in Algorithm 3, we sort the second list R' using lexicographical ordering over (π^-, π^+). Then, for each parametric interval r in R we perform a binary search on R' using $(\pi^-(r), \pi^+(r))$. After this search, using a simple iteration, we find the set of parametric intervals Y that might have a non-empty intersection with r. For concatenation, in Algorithm 4, we sort the second list R' using π^-. Then, for each parametric interval r in R we perform binary search on R' using $\pi^+(r)$. Thus, we find the set Y' of parametric intervals that start where r ends. The use of sorting combined with binary search makes the Algorithms 3,4 efficient. Hence a lot of redundant polytopic operations can be avoided. It is also important to note that the number of variables involved in the polytopes is also reduced by two (beginning and ending of intervals can be stored as constants).

Algorithm 3 INTERVAL_INTERSECTION(R, R')
assume R' sorted by (π^-, π^+) lexicographically

$R'' := \emptyset$
for each $r \in R$ **do**
  $(l, l') := (\pi^-(r), \pi^+(r))$
  $Y = \{z \in R' : (\pi^-(z) = l) \wedge (\pi^+(z) = l')\}$ ▷ using binary search
  for each $z \in Y$ **do**
    $z'' := r \cap z'$
    $R'' := R'' \cup z''$
return R''

Algorithm 4 INTERVAL_CONCAT(R, R')
assume R' sorted by π^-

```
R'' := ∅
for each r ∈ R do
    l' := π+(r)
    Y' = {z ∈ R' : π−(z) = l'}          ▷ using binary search
    for each z ∈ Y' do
        z'' := r ∘ z'
        R'' := R'' ∪ z''
return R''
```

4.3 Transitive Closure for Kleene Star

For computing the transitive closure (required for matching Kleene star) over both a union of parametric zones and a union of parametric intervals, we can directly use the squaring based algorithm given in [21] (see Appendix A), by choosing the concatenation operation specific to unions of parametric zones or of parametric intervals described in Subsections 4.1 and 4.2.

5 PARAMETRIC IDENTIFICATION FOR PSRE

In this section we deal with a PSRE parametric identification problem stated as follows: given labelled signals, find parameter values in a PSRE that produce the matches corresponding to the labels.

A computational problem arises when the expression contains atomic predicates of form $x \leq q$ or $x \geq q$ with the magnitude parameter q. For a given signal w, the parametric match-sets $\mathcal{M}(x \leq q, w)$ and $\mathcal{M}(x \geq q, w)$ contain a number of parametric zones approximately equal to the number of samples in w, which can be large in practice. Unlike STL that involves absolute time, TRE is more specific to relative time and we can thus exploit this specificity to decompose the signal horizon into non-overlapping time intervals and then perform the computation separately on each interval. To explain this idea, we first need the concept of *decisive region*.

5.1 Decisive Regions

Let us assume we are given a PSRE ψ over magnitude parameters q and timing parameters s. The expression ψ is defined over a signal w. We use vectors u and v to represent the parametric valuations for magnitude and timing parameters respectively. Given a time interval $I = [a, b]$, the decisive region $\mathcal{D}(I)$ corresponding to the interval I is defined as follows:

$$\mathcal{D}(I) := \{(t, t', u, v) : (a \leq t) \wedge (t' \leq b) \wedge (t \leq t')\}$$

The aim is to compute the intersection of the parametric match-set for ψ and w with the decisive region, that is $\mathcal{M}(\psi, w) \cap \mathcal{D}(I)$.

As an illustration we show in Figure 4 interval I_1 with the corresponding decisive region $\mathcal{D}(I_1)$ in green. Consider Interval I_{out} which is outside $\mathcal{D}(I_1)$. It goes to the left by left concatenation and upwards for right concatenation, to points I_{lc} and I_{rc} respectively. It becomes clear that the intervals outside the decisive region can never come inside it by application of concatenation operation. Therefore, we can safely ignore the points outside the decisive region.

Now, we show some equivalences that will allow us to compute the above intersection efficiently in an inductive manner. Let us

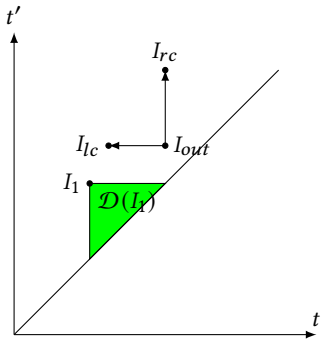


Figure 4: Points outside will be pushed further outside with concatenation

consider the PSRE φ_1,φ_2. It is easy to see that

$$\mathcal{M}(\varphi_1 \vee \varphi_2, w) \cap \mathcal{D}(I) = (\mathcal{M}(\varphi_1, w) \cap \mathcal{D}(I)) \cup (\mathcal{M}(\varphi_2, w) \cap \mathcal{D}(I));$$

$$\mathcal{M}(\varphi_1 \wedge \varphi_2, w) \cap \mathcal{D}(I) = (\mathcal{M}(\varphi_1, w) \cap \mathcal{D}(I)) \cap (\mathcal{M}(\varphi_2, w) \cap \mathcal{D}(I));$$

$$\mathcal{M}(\langle\varphi_1\rangle_I, w) \cap \mathcal{D}(I) = (\mathcal{M}(\varphi_1, w) \wedge \mathcal{D}(I)) \cap \{(t, t', u, v) : t' - t \in I_{u,v}\};$$

Now for the final case of $\varphi_1 \cdot \varphi_2$, we need to prove the equality

$$\mathcal{M}(\varphi_1 \cdot \varphi_2, w) \cap \mathcal{D}(I) = (\mathcal{M}(\varphi_1, w) \cap \mathcal{D}(I)) \circ (\mathcal{M}(\varphi_2, w) \cap \mathcal{D}(I)) \cap \mathcal{D}(I);$$

The following inclusion is straightforward:

$$(\mathcal{M}(\varphi_1, w) \cap \mathcal{D}(I)) \circ (\mathcal{M}(\varphi_2, w) \cap \mathcal{D}(I)) \cap \mathcal{D}(I) \subseteq \mathcal{M}(\varphi_1 \cdot \varphi_2, w) \cap \mathcal{D}(I);$$

Let $(t_0, t_0', u, v) \in \mathcal{M}(\varphi_1 \cdot \varphi_2, w) \cap \mathcal{D}(I)$. This implies there exists t_0'' such that $(t_0, t_0'', u, v) \in \mathcal{M}(\varphi_1)$ and $(t_0'', t_0', u, v) \in \mathcal{M}(\varphi_2)$. Since $(t_0, t_0', u, v) \in \mathcal{D}(I)$ it implies that $(t_0, t_0'', u, v) \in \mathcal{D}(I)$ and $(t_0'', t_0', u, v) \in \mathcal{D}(I)$. Therefore, the following inclusion is also true:

$$\mathcal{M}(\varphi_1 \cdot \varphi_2, w) \cap \mathcal{D}(I) \subseteq (\mathcal{M}(\varphi_1, w) \cap \mathcal{D}(I)) \circ (\mathcal{M}(\varphi_2, w) \cap \mathcal{D}(I)) \cap \mathcal{D}(I);$$

which establishes the equality we need to prove. □

Now, we consider how to efficiently compute $\mathcal{M}(\psi, w) \cap \mathcal{D}(I)$. Note that the intersection with the decisive region trickles right down to the leaves of the parse tree till it reaches the atomic predicates. Using the four aforementioned equivalences, to compute $\mathcal{M}(\psi, w) \cap \mathcal{D}(I)$ we only need to handle $\mathcal{M}(x \leq p, w) \cap \mathcal{D}(I)$. We can see now the interest of the concept of decisive region since the number of parametric zones to consider is reduced to the number of samples in the part of the signal between time units a and b i.e. in $w[a, b]$. This is particularly useful when the signal w is very large.

5.2 Parametric Identification using Decisive Regions

We consider the following scenario: signals with labelled intervals are provided to us by some expert, (e.g. temperature records and heat waves, road traffic records and dangerous overtakings, etc.). Our aim is to automatise the labelling of the signal (e.g. finding heat waves in unlabelled temperature curves) by pattern matching. To do so, we try to find (aka. identify) parameter valuations for a

given parametric formula so that matches of the formula over the signals corresponds to the labelled intervals of the expert.

Given a PSRE ψ, a signal w and a list $\mathcal{I}$ of n intervals $I_1 = [a_1, b_1], \ldots, I_n = [a_n, b_n]$, we want to compute the set of parameters that produce a match at each of these n intervals that is the set $\mathcal{P}(\psi, w, \mathcal{I})$ of all (u, v) such that

$$\bigwedge_{1 \le k \le n} (\exists t, t' \ (t, t', u, v) \in \mathcal{M}(\psi, w) \wedge t = a_k \wedge t' = b_k)\}$$

We call $\mathcal{P}(\psi, w, \mathcal{I})$ the solution set of ψ, w and $\mathcal{I}$. Note that $(t, t', u, v) \in \mathcal{M}(\psi, w) \wedge t = a_k \wedge t' = b_k$ is equivalent to $(t, t', u, v) \in (\mathcal{M}(\psi, w) \cap \mathcal{D}(I_k)) \wedge t = a_k \wedge t' = b_k$. Therefore, the previously described concept of decisive region can be applied for parametric identification.

5.3 Combining Booleanization and Parametric Matching

While the concept of decisive region can help reducing the number of parametric zones to be handled at a time, this number can still be large, often due to the magnitude parameters. In order to further reduce time complexity, it is of interest to separate magnitude parameters and perform matching expressions containing only timing parameters. This can be done by Booleanizing the signals where the Booleanization involves magnitude variables. Such Booleanization can be done using various operations over signals. In this work, we illustrate the use of *extended STL* [8] which is particularly appropriate for this purpose since this formalism allows expressing quantitatively shapes of signals. Indeed, combining with this specification language we can avoid the use of magnitude parameters in several cases. For instance to detect a peak with unknown height one could be tempted to use a magnitude parameter θ and write $x \ge \theta$. With extended STL it suffices to detect the maximum over a window as illustrated below. Another example is the stabilisation of a signal around an unknown value θ within a tolerance ϵ, that is, $\theta - \epsilon \le x \le \theta + \epsilon$. We can avoid the use of the magnitude parameter θ by saying that the difference between the maximum and minimum on a window I is within the tolerance ϵ, that is, $\max_I x - \min_I x \le \epsilon$.

In Figure 2, the ECG pulse is shown as having a maximum between two minima. We can Booleanize an ECG signal x to get two Boolean signals b_{max}, b_{min} for the approximate maximum and minimum respectively using the operators defined in [8] as below:

$$(b_{max} := \max_{[-150,150]} x - x \le 0.05), \ (b_{min} := x - \min_{[-10,10]} x \le 0.05) \tag{5}$$

The Boolean signals b_{max} and b_{min} are true at time points where there are approximate maxima and minima respectively. More precisely, b_{max} is true at time points where the current value is within 0.05 of the maximum value in the time interval $[-150, 150]$ around it. Similarly, b_{min} is true at time points where the current value is within 0.05 of the minimum value in the time interval $[-10, 10]$ around it. Now we have an abstraction of an ECG in the form of ϕ_{10} (see Expr (6)).

$$\phi_{10} := b_{min} \cdot \langle \mathtt{true} \rangle_{[0,\theta_1]} \cdot b_{max} \cdot \langle \mathtt{true} \rangle_{[0,\theta_2]} \cdot b_{min} \tag{6}$$

6 EXPERIMENTAL RESULTS

We implemented the algorithms discussed in Section 4 in C++, into a tool named `parameTRE`. We represent parametric zones as polytopes. Parametric intervals are represented as a specialized C++ class with the beginning and end of the interval stored as member variables. Constraints on parameters are stored as a member polytope. For PSRE and PE-TRE, we maintain a parametric match set as a union of polytopes. For event-based PTRE, we maintain it as a union of parametric intervals. To handle polytopes, the tool uses the Parma Polyhedra Library (PPL) [7]. We evaluate the performance of the tool using various kinds of expressions, signals and timed words. All experiments have been performed on a laptop with a Core i7-8665U and 16GB RAM. We first show experimental results for synthetic examples to evaluate the correctness of the implementation and the performance of the proposed algorithms. Then, we show how we can match or detect interesting behaviours in real-life scenarios.

6.1 Synthetic Examples

We first describe the data (signals and timed words) and expressions we are dealing with and then give the experimental results.

6.1.1 Signals and Timed Words. We use four different types of signals and a single timed word. The signal $wsynth_1$ for ϕ_1 (Expr (1)) consists of signal p which is a square wave of period 14 and q which is p shifted by 3 time units. Both p and q are repeated 1000 times. The signal $wbrake$ for ϕ_4 (Expr (4)) consists of the signal b which represents the braking pattern. The signal $wsynth_2$ for ϕ_5 (Expr (7)) consists of signal p_0 which is a square wave of period 200 time units. We get p_1 and p_2 by shifting p_0 by 35 and 45 time units respectively. All of p_0, p_1 and p_2 are repeated 1000 times. The timed word $word_1$ for ϕ_6 (Expr (8)) consists of alternating occurrences of b and a with a total of around 1000 events.

6.1.2 Expressions. The expressions ϕ_1 and ϕ_4 have already been discussed in Sections 1 and 3. The expressions ϕ_5 and ϕ_6 given below are simple examples of Kleene closure operation for state-based and event-based paradigms respectively.

$$\phi_5 := p_0 \cdot (\langle p_1 \rangle_{\theta_1} \cdot \langle p_2 \rangle_{\theta_2})^+ \tag{7}$$

$$\phi_6 := \langle \underline{a} \cdot \langle \underline{b} \cdot \underline{a} \rangle^*_{[\theta_2,\theta_3]} \cdot \underline{b} \rangle_{\theta_1} \tag{8}$$

The experimental results are summarized in Table 1. The column "Data" gives the type of the signal or timed word. The column "Size" expresses the number of samples for signals and number of events for timed words. The column "Matches" expresses the number of parametric zones in a match-set. The "Time" column gives the total time taken to find matches in the given data. In Table 1 and also in Table 2, we mentioned the number of parametric zones/intervals in the column "Matches" since it can serve as an empirical measure of geometric complexity of match sets.

A parametric zone in the parametric match set for ϕ_1 and $wsynth_1$ is: $(t' - \theta_2 = 3) \wedge (t + \theta_1 = 7) \wedge (\theta_3 \le 4) \wedge (t' \le 10) \wedge (t \le 3) \wedge (\theta_3 \ge 1) \wedge (t \ge 0) \wedge (t' \ge 7)$.

Another parametric zone which comes from the parametric match set for ϕ_5 and $wsynth_2$: $(\theta_1 \ge 1) \wedge (\theta_2 \ge 1) \wedge (\theta_2 \le 100) \wedge (t \ge 20) \wedge (t' \le 200) \wedge (t' - t - \theta_1 - \theta_2 \ge 0) \wedge (t' - \theta_2 \le 135) \wedge (t' - \theta_1 - \theta_2 \le 120) \wedge (t' - \theta_2 \ge 45) \wedge (t' - \theta_1 - \theta_2 \ge 35)$.

Expression	Data	Size	Matches	Time
ϕ_1 (Expr 1)	$wsynth_1$	4000	1000	90s
ϕ_4 (Expr 4)	$wbrake$	16	9	1.3s
ϕ_5 (Expr 7)	$wsynth_2$	6000	8000	87s
ϕ_6 (Expr 8)	$word_1$	1002	125251	77s

Table 1: Experiments With Synthetic Data

6.2 Real-Life Scenarios

We now describe how we can use PSRE to find matches for real-life behaviours of interest. Electrocardiogram is a simple test that can be used to check the heart's rhythm and electrical activity. The signals $wecg_{205}$, $wecg_{221}$ and $wecg_{123}$ correspond to the Electrocardiograms (ECG) 205, 221 and 123 respectively taken from the MIT-BIH Arrhythmia Database [16, 20]. First, we consider the problems of matching and parametric identification for ECG pulses. Then, we show how we can utilize Booleanization of signals to aid with PSRE matching. Finally, we describe how we can detect marine traffic rule violations involving ships crossing each other.

6.2.1 Matching ECG Pulses Using Expression With Only Timing Parameters. Consider ϕ_2 (Expr (2)) that has only timing parameters. The ECG signals are denoted by the symbol x in ϕ_2. The experimental results for matching of ϕ_2 for the three ECG signals are in Table 2. The "Error" column in Table 2 gives an estimate of the error involved when detecting ECG pulses using ϕ_2 when compared to an expert doctor.

Expression	Data	Size	Matches	Time	Error
ϕ_2 (Expr 2)	$wecg_{205}$	650000	2646	9s	2.83%
ϕ_2 (Expr 2)	$wecg_{221}$	650000	2638	9s	23%
ϕ_2 (Expr 2)	$wecg_{123}$	650000	1518	9s	0.2%

Table 2: ECG Matching Experiments

A parametric zone in the parametric match set for ϕ_2 and $wecg_{205}$ is: $(t + \theta_1 = 227) \wedge (\theta_2 = 6) \wedge (t' - \theta_3 = 233) \wedge (220 \leq t \leq 226) \wedge (234 \leq t' \leq 283)$

Projecting it on to the parameter space gives the following rectangle: $(1 \leq \theta_1 \leq 7) \wedge (\theta_2 = 6) \wedge (1 \leq \theta_3 \leq 50)$. Figure 5 shows an illustration for ϕ_2 and $wecg_{205}$. We take two points from the aforementioned rectangle and plot the corresponding matches in the signal.

6.2.2 Parametric Identification for Labelled ECG-205 (Mini). Let us consider the sub-signal $wecg_{mini205}$ of $wecg_{205}$ containing the first ten pulses. $wecg_{mini205}$ has the size of 2500 time units. For each of these ten pulses we are given ten intervals of length 80 time units. These intervals are denoted by $\mathcal{I}_{205}$. The expressions ϕ_7, ϕ_8 and ϕ_9 (Expressions 9,10,11) have 4, 5 and 6 parameters respectively some of them being magnitude parameters. We would like to compute the solution sets for these three expressions with respect to $wecg_{mini205}$ and $\mathcal{I}_{205}$. The solution sets for all expressions contain exactly 75 polytopes. The computation times for ϕ_7, ϕ_8 and ϕ_9 are 478s, 888s and 1881s respectively and increase exponentially with the number of parameters. The original ECG-205 contains around 2000 pulses and it becomes intractable to do parametric identification for all these 2000 pulses. Therefore, in the next subsection, to reduce the time complexity, we use Booleanization of signals as a preprocessing step, as shown in Subsection 5.3

$$\phi_7 := \langle -q_0 \leq x \leq q_1 \rangle_{[0,q_2]} \cdot \langle q_1 \leq x \leq q_0 \rangle_{[0,q_3]} \cdot \langle -q_0 \leq x \leq q_1 \rangle_{[0,q_2]} \tag{9}$$

$$\phi_8 := \langle -q_0 \leq x \leq q_1 \rangle_{[0,q_3]} \cdot \langle q_1 \leq x \leq q_2 \rangle_{[0,q_4]} \cdot \langle -q_0 \leq x \leq q_1 \rangle_{[0,q_3]} \tag{10}$$

$$\phi_9 := \langle -q_0 \leq x \leq q_1 \rangle_{[0,q_3]} \cdot \langle q_1 \leq x \leq q_2 \rangle_{[0,q_4]} \cdot \langle -q_0 \leq x \leq q_1 \rangle_{[0,q_5]} \tag{11}$$

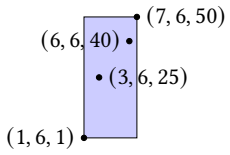
6.2.3 Booleanization and Matching for ECGs. Booleanization described by (5) in Subsection 5.3 has been used to obtain the Boolean signals b_{min} and b_{max}. We do matching using ϕ_{10} (Expr (6)) in which the time delay between the first minimum and the maximum is bounded by the parameter $\theta_1 \in [0, 20]$. And the time delay between the maximum and the second minimum is bounded by $\theta_2 \in [0, 20]$. Note that we have multiple matches per pulse, and the number of computed matches of ϕ_{10} with respect to ECGs 205, 221 and 123 are 8726, 8144 and 1971 respectively computed under 33s, 51s and 29s respectively.

6.2.4 Detecting Marine Traffic Rule Violations on Florida Coast. Here, we deal with finding marine traffic rule violations when two ships are crossing each other. We use scenarios from Marine Cadastre dataset[1] preprocessed[2] by [17]. From the preprocessed data we generate Boolean signals denoted by propositional variables p_1 and p_2. The variable p_1 is true at time points when the ego ship detects a crossing with another ship. The variable p_2 is true at time points where a maneuver appropriate for crossing scenario is executed.

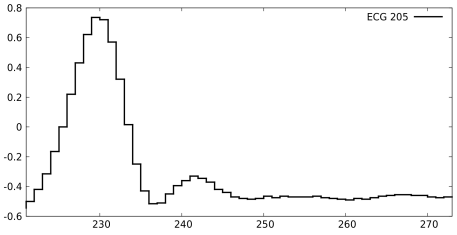
$$\phi_{11} := \overline{p_1} \cdot (\langle \mathtt{true} \rangle_{dt} \cdot \langle p_1 \rangle_{\theta_r - dt} \cdot \mathtt{true} \wedge \langle \overline{p_2} \rangle_{\theta_r + \theta_m} \cdot \mathtt{true}) \vee \overline{p_1} \cdot (\langle \mathtt{true} \rangle_{dt} \cdot \langle p_1 \rangle_{\theta_r - dt} \cdot \mathtt{true} \wedge \langle \mathtt{true} \rangle_{\theta_r} \cdot \langle p_1 \rangle_{2\theta_m} \cdot \mathtt{true}) \tag{12}$$

We formalise traffic rule violation for crossing ships as matching of ϕ_{11} (Expr (12)) where the constant $dt = 1$ is the time step, parameter $\theta_r \in [6, 20]$ is the reaction time and $\theta_m \in [6, 20]$ is the maneuver time. The expression captures two kinds of rule violations. The first being that a crossing detection is maintained until θ_r time and no maneuver is observed for the duration of the sum of reaction time and maneuver time. The second being that a crossing is detected like before but the crossing situation is maintained for the duration of twice the maneuver time. The formulation has been heavily inspired by the STL formula for traffic rule R_3 (related to crossing ships) in Table 1 of [17]. We introduce parameters θ_r and θ_m as they can serve to estimate the reaction and maneuver time while matching. We did matching for 370 signals obtained from scenarios on the Florida coast. Each signal contains two components corresponding to crossing and maneuvering represented by p_1 and p_2 respectively. Each signal contains between 100 to 500 time steps. Out of the 370 signals, we detected a violation/matching

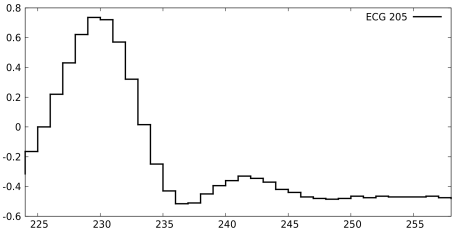
[1] https://marinecadastre.gov/ais/
[2] doi.org/10.24433/CO.8258454.v2



(a) Parameter set (rectangle) from matching of ϕ_2 and $wecg_{205}$



(b) $(t, t') = (221, 273)$ **for** $(\theta_1, \theta_2, \theta_3) = (6, 6, 40)$



(c) $(t, t') = (224, 258)$ **for** $(\theta_1, \theta_2, \theta_3) = (3, 6, 25)$

Figure 5: Matching ECG-205 signal ($wecg_{205}$) with ϕ_2

in one signal and none in the others. The detected violation involves a crossing situation of duration 5 time units followed by no maneuver for more than 12 time units. This violation is found in the scenario with the tag USOCEAN_Florida-20190103_7_T-1 and occurs at around 98 time units from the start of the signal. The matching time combined for all the 370 signals took around 1.26 seconds.

7 RELATED WORK

The closest work to ours is on PSTL and PTA. First, we compare with PSTL. We have seen in Subsection 6.2.2 that exact parametric identification for PSRE containing both magnitude and timing parameters is intractable except for very small signals. Similarly, for PSTL, solving the parametric identification problem exactly is insurmountably difficult except for small signals. For PSTL, to reduce the complexity of the problem, two alternative approaches have been explored. The first approach involves making the assumption that the PSTL formulae are monotonic and exploit it to efficiently compute approximate validity domains as in [6, 13]. The second approach is to restrict the focus to formulae with only magnitude parameters and utilize specialized algorithms as in [9]. Both these approaches can be modified and applied also to PSRE but we leave this to future work. A comprehesive survey on methods for parametric identification for PSTL and general approaches that learn the structure as well as parameter values of STL formulae can be found in [12].

Now, we compare with PTA. Similar to ϕ_6 (Expr (8)) is the BLOWUP PTA from [3]. For this example, using parametric intervals, we could outperform their approach. PTRE matching takes 77s whereas matching with BLOWUP PTA has been reported to take 940.74s in [3]. PTRE (event-based) is strictly less expressive than PTA. On the other hand, parametric timed pattern matching of [25] does not handle state-based semantics and magnitude parameters while PSRE does. We defer to future work, automated translation and comprehensive comparison of PTRE and PTA. If we consider event-based semantics, matching is faster using PTRE than PTA. The reason is that handling parametric intervals with sorting and binary search in the matching algorithms is more time-efficient than handling parameters and variables in the same polytopic space. Finally, in terms of usage, PSTL and PTA have their own advantages. PSTL easily supports notions of false positive and negatives while PTA is more expressive than PTRE.

8 CONCLUSION AND FUTURE WORK

In this paper, we explore parametric versions of different types of timed regular expressions. We show that the parametric match sets can be computed and represented using parametric zones and parametric intervals, which can be seen as two particular types of polytopes that can be handled more efficiently than general polytopes. Indeed, the constraints over time variables in these polytopes have a simple form. To perform operations on unions of parametric zones, we slightly modify the existing algorithms for non-parametric case. For operations on unions of parametric intervals, we devise efficient algorithms that have a number of polyhedral operations either log-linear in input or linear in output. As experimental results, we first deal with various synthetic examples. And then, we present real-life scenarios involving matching/parametric identification of Electrocardiogram (ECG) pulses and detecting marine traffic rule violations by ships.

As future work, atomic predicates involving integrals of the form $\int_t^{t'} x.dt \leq c$ can be introduced. Consider the expression $\langle\varphi\rangle_{[2-\delta,3]}$, we can examine how inserting timing parameters like δ is related to measuring timed robustness [10]. Additionally, we can restrict the scope to expressions with only magnitude parameters and efficiently handle them using a data structure that combines zones and boxes. Finally, we can explore how parametric identification of PSRE with monotonicity property can be efficiently performed using queries as it has been done for PSTL in [6, 13].

ACKNOWLEDGMENTS

This work was financed by the ANR MAVeriQ (ANR-20-CE25-0012), the joint ANR-JST project CyPhAI.

REFERENCES

[1] Rajeev Alur and David L. Dill. 1994. A Theory of Timed Automata. *Theoretical Computer Science* 126, 2 (April 1994), 183–235.

[2] Rajeev Alur, Thomas A. Henzinger, and Moshe Y. Vardi. 1993. Parametric real-time reasoning. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, May 16-18, 1993*, S. Rao Kosaraju, David S. Johnson, and Alok Aggarwal (Eds.). ACM, San Diego, CA, USA, 592–601. https://doi.org/10.1145/167088.167242

[3] Étienne André, Ichiro Hasuo, and Masaki Waga. 2018. Offline Timed Pattern Matching under Uncertainty. In *23rd International Conference on Engineering of Complex Computer Systems, ICECCS 2018, December 12-14, 2018*. IEEE Computer Society, Melbourne, Australia, 10–20. https://doi.org/10.1109/ICECCS2018.2018.00010

[4] Eugene Asarin, Paul Caspi, and Oded Maler. 1997. A Kleene Theorem for Timed Automata. In *Proceedings, 12th Annual IEEE Symposium on Logic in Computer Science, June 29 - July 2, 1997*. IEEE Computer Society, Warsaw, Poland, 160–171. https://doi.org/10.1109/LICS.1997.614944

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

[6] Eugene Asarin, Alexandre Donzé, Oded Maler, and Dejan Nickovic. 2011. Parametric Identification of Temporal Properties. In *Runtime Verification - Second International Conference, RV 2011, September 27-30, 2011, Revised Selected Papers (Lecture Notes in Computer Science, Vol. 7186)*, Sarfraz Khurshid and Koushik Sen (Eds.). Springer, San Francisco, CA, USA, 147–160. https://doi.org/10.1007/978-3-642-29860-8_12

[7] Roberto Bagnara, Patricia Hill, and Enea Zaffanella. 2008. The Parma Polyhedra Library: Toward a Complete Set of Numerical Abstractions for the Analysis and Verification of Hardware and Software Systems. *Science of Computer Programming* 72 (06 2008), 3–21. https://doi.org/10.1016/j.scico.2007.08.001

[8] Alexey Bakhirkin and Nicolas Basset. 2019. Specification and Efficient Monitoring Beyond STL. In *Tools and Algorithms for the Construction and Analysis of Systems - 25th International Conference, TACAS 2019, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2019, April 6-11, 2019, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 11428)*, Tomás Vojnar and Lijun Zhang (Eds.). Springer, Prague, Czech Republic, 79–97. https://doi.org/10.1007/978-3-030-17465-1_5

[9] Alexey Bakhirkin, Thomas Ferrére, and Oded Maler. 2018. Efficient Parametric Identification for STL. In *Proc. 21st Int. Conf. on Hybrid Systems: Computation and Control (HSCC'18)*. ACM, New York, NY, USA, 177–186.

[10] Alexey Bakhirkin, Thomas Ferrère, Oded Maler, and Dogan Ulus. 2017. On the Quantitative Semantics of Regular Expressions over Real-Valued Signals. In *FORMATS (Lecture Notes in Computer Science, Vol. 10419)*. Springer, Berlin, Germany, 189–206.

[11] Alexey Bakhirkin, Thomas Ferrère, Dejan Nickovic, Oded Maler, and Eugene Asarin. 2018. Online Timed Pattern Matching Using Automata. In *FORMATS (Lecture Notes in Computer Science, Vol. 11022)*. Springer, Beijing, China, 215–232.

[12] Ezio Bartocci, Cristinel Mateis, Eleonora Nesterini, and Dejan Nickovic. 2022. Survey on mining signal temporal logic specifications. *Inf. Comput.* 289, Part (2022), 104957.

[13] Nicolas Basset, Thao Dang, Akshay Mambakam, and José-Ignacio Requeno Jarabo. 2020. Learning Specifications for Labelled Patterns. In *FORMATS (Lecture Notes in Computer Science, Vol. 12288)*. Springer, Vienna, Austria, 76–93.

[14] Alexandre Donzé and Oded Maler. 2010. Robust Satisfaction of Temporal Logic over Real-Valued Signals. In *Formal Modeling and Analysis of Timed Systems - 8th International Conference, FORMATS 2010, September 8-10, 2010. Proceedings (Lecture Notes in Computer Science, Vol. 6246)*, Krishnendu Chatterjee and Thomas A. Henzinger (Eds.). Springer, Klosterneuburg, Austria, 92–106. https://doi.org/10.1007/978-3-642-15297-9_9

[15] Thomas Ferrère, Oded Maler, Dejan Nickovic, and Dogan Ulus. 2015. Measuring with Timed Patterns. In *CAV (2) (Lecture Notes in Computer Science, Vol. 9207)*. Springer, San Francisco, CA, USA, 322–337.

[16] Ary L. Goldberger, Luis A.N. Amaral, Leon Glass, Jeffrey M. Hausdorff, Plamen Ch. Ivanov, Roger G. Mark, Joseph E. Mietus, George B. Moody, Chung-Kang Peng, and H. Eugene Stanley. 2000. PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals. *Circulation* 101, 23 (2000), e215–e220.

[17] Hanna Krasowski and Matthias Althoff. 2021. Temporal Logic Formalization of Marine Traffic Rules. In *IEEE Intelligent Vehicles Symposium, IV 2021, July 11-17, 2021*. IEEE, Nagoya, Japan, 186–192. https://doi.org/10.1109/IV48863.2021.9575685

[18] Oded Maler and Dejan Nickovic. 2004. Monitoring Temporal Properties of Continuous Signals. In *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems, Joint International Conferences on Formal Modelling and Analysis of Timed Systems, FORMATS 2004 and Formal Techniques in Real-Time and Fault-Tolerant Systems, FTRTFT 2004, September 22-24, 2004, Proceedings (Lecture Notes in Computer Science, Vol. 3253)*, Yassine Lakhnech and Sergio Yovine (Eds.). Springer, Grenoble, France, 152–166. https://doi.org/10.1007/978-3-540-30206-3_12

[19] Oded Maler, Dejan Nickovic, and Amir Pnueli. 2006. From MITL to Timed Automata. In *FORMATS (Lecture Notes in Computer Science, Vol. 4202)*. Springer, Paris, France, 274–289.

[20] George B Moody and Roger G Mark. 2001. The impact of the MIT-BIH Arrhythmia Database. *IEEE Engineering in Medicine and Biology Magazine* 20, 3 (2001), 45–50.

[21] Dogan Ulus, Thomas Ferrère, Eugene Asarin, and Oded Maler. 2014. Timed Pattern Matching. In *FORMATS (Lecture Notes in Computer Science, Vol. 8711)*. Springer, Florence, Italy, 222–236.

[22] Dogan Ulus and Oded Maler. 2018. Specifying Timed Patterns using Temporal Logic. In *HSCC*. ACM, Porto, Portugal, 167–176.

[23] Masaki Waga and Étienne André. 2019. Online Parametric Timed Pattern Matching with Automata-Based Skipping. In *NFM (Lecture Notes in Computer Science, Vol. 11460)*. Springer, Houston, TX, USA, 371–389.

[24] Masaki Waga, Étienne André, and Ichiro Hasuo. 2019. Symbolic Monitoring Against Specifications Parametric in Time and Data. In *CAV (1) (Lecture Notes in Computer Science, Vol. 11561)*. Springer, New York City, NY, USA, 520–539.

[25] Masaki Waga, Étienne André, and Ichiro Hasuo. 2023. Parametric Timed Pattern Matching. *ACM Trans. Softw. Eng. Methodol.* 32, 1, Article 10 (feb 2023), 35 pages. https://doi.org/10.1145/3517194

A TRANSITIVE CLOSURE

Here, we present the incremental and squaring methods for computing the transitive closure in Algorithms 5 and 6 respectively.

Algorithm 5 CLOSURE(Z) *incremental*

$X := Z$
$Y := \text{CONCAT}(Z, Z)$
while $Y \not\sqsubseteq X$ **do**
  $X := Y \cup X$
  $Y := \text{CONCAT}(Y, Z)$
return X

Algorithm 6 CLOSURE2(Z) *squaring*

$X := Z$
$Y := \text{CONCAT}(Z, Z)$
while $Y \not\sqsubseteq X$ **do**
  $X := Y \cup X \cup \text{CONCAT}(Y, X)$
  $Y := \text{CONCAT}(Y, Y)$
return X

B PROOFS OF CHARACTERIZATION OF PARAMETRIC MATCH-SETS

Theorem (Theorem 15 (recap)). *For a timed word the parametric match-set of a PTRE is a finite union of parametric intervals.*

Proof. Let $\mathbf{y}_1 := (t = d_1 \wedge t' = d_2 \wedge \boldsymbol{C_{y_1}})$ and $\mathbf{y}_2 := (t = e_1 \wedge t' = e_2 \wedge \boldsymbol{C_{y_2}})$ be two parametric intervals where d_1, d_2, e_1, e_2 are constants. We prove the theorem by considering the various cases in the definition of PTRE.

For $\underline{a}$ and ϵ, one can see that there exist a finite number of intervals in the match set.

If we apply duration restriction operation $\langle\varphi\rangle_{[\alpha,\beta]}$ on $\mathbf{y}_1$ the resulting parametric interval $\mathbf{y}$ can be written as follows,

$\mathbf{y} := (t = d_1 \wedge t' = d_2 \wedge \alpha \le t' - t \le \beta \wedge \boldsymbol{C_{y_1}})$

It can be further simplified as,

$\mathbf{y} := (t = d_1 \wedge t' = d_2 \wedge (\alpha \le d_2 - d_1 \le \beta \wedge \boldsymbol{C_{y_1}}))$

Finite unions of parametric intervals are closed under Boolean and concatenation operations. For concatenation operation $\varphi \cdot \psi$, let us consider the sequential composition $\mathbf{y} := \mathbf{y}_1 \circ \mathbf{y}_2$,

$\mathbf{y} := \exists t'' \cdot (t = d_1 \wedge t'' = d_2 \wedge \boldsymbol{C_{y_1}}) \wedge (t'' = e_1 \wedge t' = e_2 \wedge \boldsymbol{C_{y_2}})$

The resulting parametric interval $\mathbf{y}$ is non-empty only when $d_2 = e_1$. We can further simplify the equation for the resulting $\mathbf{y}$ as follows:

$\mathbf{y} := (t = d_1 \wedge t' = e_2 \wedge (\boldsymbol{C_{y_1}} \wedge \boldsymbol{C_{y_2}}))$

For intersection operation $\varphi \wedge \psi$, let us consider $\mathbf{y} := \mathbf{y}_1 \cap \mathbf{y}_2$, i.e.,

$\mathbf{y} := (t = d_1 \wedge t' = d_2 \wedge \boldsymbol{C_{y_1}}) \wedge (t = e_1 \wedge t' = e_2 \wedge \boldsymbol{C_{y_2}})$

The resulting parametric interval $\mathbf{y}$ is non-empty only when $d_1 = e_1$ and $d_2 = e_2$. For this case, we can further simplify the equation for $\mathbf{y}$ as follows: $\mathbf{y} := (t = d_1 \wedge t' = d_2 \wedge (\boldsymbol{C_{y_1}} \wedge \boldsymbol{C_{y_2}}))$

For union operation $\varphi \vee \psi$, we simply concatenate the two lists of parametric intervals that correspond to the expressions.

For the case of Kleene star the reasoning to show that the match set is finite union of parametric intervals is as follows. Let us consider a timed word $\omega = t_1 a_1 ... t_n a_n$. From the semantics in Definition 14 it follows that for any parametric interval $y_1 := (t = d_1 \wedge t' = d_2 \wedge \boldsymbol{C_{y_1}})$ the start value $t = d_1$ and the end value $t = d_2$ both take values only from $\{t_1, ..., t_n\}$. They cannot take values in the dense space in between these discrete points. From this it follows that the number of possible values for (t, t') in the parametric match-set is at most quadratic in n. The exception is when one of the parametric intervals involved in sequential composition is of time length zero. Let us assume that in the sequential composition, the first parametric interval is $y_1 := (t = d_1 \wedge t' = d_2 \wedge \boldsymbol{C_{y_1}})$ and the second one $y_2 := (t = d_2 \wedge t' = d_2 \wedge \boldsymbol{C_{y_2}})$ is of time length zero. It follows that $\mathbf{y} := \mathbf{y}_1 \circ \mathbf{y}_2 := (t = d_1 \wedge t' = d_2 \wedge (\boldsymbol{C_{y_1}} \wedge \boldsymbol{C_{y_2}}))$. One can notice that $\mathbf{y} \subseteq \mathbf{y}_1$. So, $\mathbf{y}$ can be safely ignored when computing the transitive closure. Similar reasoning also applies when $\mathbf{y}_1$ is of time length zero. Therefore, the parametric match-set for Kleene star contains a finite number of parametric intervals. □

Theorem (Theorem 16 (recap)). *For Boolean, piecewise linear and piecewise constant signals the parametric match-set of PE-TRE is always a finite union of parametric intervals.*

Proof. Let $\mathbf{y}_1 = (t = d_1 \wedge t' = d_2 \wedge \boldsymbol{C_{y_1}})$ and $\mathbf{y}_2 = (t = e_1 \wedge t' = e_2 \wedge \boldsymbol{C_{y_2}})$ be two parametric intervals where d_1, d_2, e_1, e_2 are constants. Let $\mathbf{z} = (t \prec c_1) \wedge (t' \prec c_2) \wedge (t' - t \prec c_3) \wedge (c_4 \prec t' - t) \wedge (c_5 \prec t') \wedge (c_6 \prec t) \wedge \boldsymbol{C_z}$ be a parametric zone. We prove the theorem by considering the various cases in the definition of PE-TRE.

For $\uparrow p$, given a Boolean signal p, there exists a finite number of time points with a rising edge. One can see that these time points can be represented as parametric intervals.

For $\psi_1 \cdot \varphi \cdot \psi_2$, we iterate and perform sequential composition operation over $\mathbf{y}_1 \in \mathcal{M}(\psi_1, w)$, $\mathbf{z} \in \mathcal{M}(\varphi, w)$ and $\mathbf{y}_2 \in \mathcal{M}(\psi_2, w)$, where w is a signal, $\mathbf{y}_1$,$\mathbf{y}_2$ are parametric intervals and $\mathbf{z}$ is a parametric zone. The sequential composition $\mathbf{y}_1 \circ \mathbf{z} \circ \mathbf{y}_2$ can be written as:

$\exists s, s' \cdot t = d_1 \wedge s = d_2 \wedge \boldsymbol{C_{y_1}} \wedge (s \prec c_1) \wedge (s' \prec c_2) \wedge (s' - s \prec c_3) \wedge (c_4 \prec s' - s) \wedge (c_5 \prec s') \wedge (c_6 \prec s) \wedge \boldsymbol{C_z} \wedge s' = e_1 \wedge t' = e_2 \wedge \boldsymbol{C_{y_2}}$

One can see that, after performing quantifier elimination over s and s' the result will be a parametric interval.

For $\psi_1 \cup \psi_2$, we simply concatenate the two lists of parametric intervals that correspond to ψ_1 and ψ_2. And for $\psi_1 \cap \varphi$, we need to perform intersection of a parametric interval with a parametric zone. One can easily see that the result of this intersection is also a parametric interval. □