A Logical Characterization of Timed (non-)Regular Languages*

Marcello M. Bersani[1], Matteo Rossi[1] and Pierluigi San Pietro[1,2]

[1] Dipartimento di Elettronica Informazione e Bioingegneria, Politecnico di Milano
[2] CNR IEIIT-MI, Milano, Italy
{marcellomaria.bersani,matteo.rossi,pierluigi.sanpietro}@polimi.it

Abstract. CLTLoc (Constraint LTL over clocks) is a quantifier-free extension of LTL allowing variables behaving like clocks over real numbers. CLTLoc is in PSPACE [9] and its satisfiability can polynomially be reduced to a SMT problem, allowing a feasible implementation of a decision procedure. We used CLTLoc to capture the semantics of metric temporal logics over continuous time, such as Metric Interval Temporal Logic (MITL), resulting in the first successful implementation of a tool for checking MITL satisfiability [7]. In this paper, we assess the expressive power of CLTLoc, by comparing it with various temporal formalisms over dense time. When interpreted over timed words, CLTLoc is equivalent to Timed Automata. We also define a monadic theory of orders, extending the one introduced by Kamp, which is expressively equivalent to CLTLoc. We investigate a decidable extension with an arithmetical next operator, which allows the expression of timed non-ω-regular languages.

1 Introduction

Linear Temporal Logic (LTL) is one of the most popular descriptive languages for modeling temporal behavior. Its time model is the structure $(\mathbb{N}, <)$, allowing the expression of positional orders of events, e.g., "if a query is received, then a reply will be delivered within 5 *positions* from now", but now allowing the formulation of real time constraints, that typically require a dense time domain. The absence of real time constitutes a major limitation of LTL, which has been addressed by adding variables, primitive operations or suitable modalities embedding real time, e.g., [19,12]. The reference model in this field is MTL (Metric Temporal Logic) [17,3], an extension of LTL that allows a temporal modality $\mathbf{U}_I$ (and $\mathbf{S}_I$), over a real time interval I. On a dense time domain, both satisfiability and model checking for MTL are undecidable [4], but various decidable fragments have been defined. MITL [2] restricts intervals I in $\mathbf{U}_I$ to be non punctual, e.g., a punctual eventuality such as $true\mathbf{U}_{[a,a]}\phi$ is not allowed. MITL is EXPSPACE-complete and it is closed under all Boolean operations. A smaller fragment of MTL, called QTL [15], is obtained by restricting

* Work supported by the Programme IDEAS-ERC, Project 227977-SMScom, and by PRIN Project 2010LYA9RH-006.

the temporal modalities to $\mathbf{U}_{(0,\infty)}$ and to $\mathbf{F}_{(0,1)}$, but it is actually equivalent to MITL. For many years, punctual eventualities were considered the main source of undecidability over dense time. In [10,20] this result has been revised: decidability has been shown not to solely depend on banning punctual intervals. For instance, Flat-MTL [10] and Safety MTL [20] allow both punctual eventualities and invariance properties, but they add some syntactical restrictions on the Until modality. They are not closed under negation, but their satisfiability is decidable. However, satisfiability of Safety-MTL formulae is non-elementary, while for Flat-MTL is EXPSPACE-complete. The dual version of Flat-MTL (i.e., consisting of the negation of formulae in Flat-MTL) is called coFlat-MTL: unfortunately, its satisfiability is undecidable (although model checking is EXPSPACE-complete).

The above research on MTL fragments, although also considered satisfiability, was mainly focused on devising logics that are suitable for model-checking of timed automata, while the interest in satisfiability has been quite limited. On the other hand, the need for full descriptive formalisms specifying reactive systems, and possibly with tractable complexity, is widely accepted [21].

Operational models, such as Büchi Automata (BA), are the most adopted and widely used alternative to temporal logic. Timed Automata (TA) [1] are the standard operational formalism for real time modeling. In [24], Büchi's famous result about the equivalence of Monadic Second-order Logic (MSO) and BA automata was extended to real time, by showing that the Monadic second-order logic $\mathcal{L}^d$ (augmenting MSO with a function measuring time between positions) is equivalent to TA. Because of its undecidability, MTL is not the proper logical formalism to capture TA, whose emptiness is decidable. Also, MTL does not embed explicit clocks, which are instead essential resources in TA: their absence makes the relationship between the expressiveness of clock constraints in TA and the syntactical restrictions of the various MTL fragments far from being evident.

In general, real time logics such as MTL are well suited to be interpreted with the *continuous-time semantics*, where atomic formulae are interpreted as state predicates, i.e., continuous flows or signals (i.e., mappings associating values in $\mathbb{R}_+$ with states). On the other hand, TA are naturally defined on the *pointwise semantics*, where atomic formulae are interpreted as instantaneous events associated with a timestamp, hence leading to an interpretation over timed words (sequences of timestamped events). TA can precisely be captured over the continuous-time semantics by various logic formalisms, such as quantified MITL [13] and Second-order real-time Sequential Calculus [2]. However, no temporal logic has so far been shown equivalent to TA in the pointwise semantics, where the construction of [2] cannot be applied.

In this paper, we bridge the gap between TA and temporal logic over the pointwise semantics. We consider Constraint LTL over clocks (CLTLoc), a quantifier-free extension of LTL that still considers discrete positions, but it has also a finite set of variables over a dense time domain, behaving like clocks of TA, to measure time elapsing among events occurring at discrete positions. Unlike MTL, clocks are explicit resources in CLTLoc and, as in TA, they can be compared with constants over $\mathbb{N}$ (or $\mathbb{Q}$). In [7], we prove that satisfiability of

CLTLoc is PSPACE-complete, by combining results on the decidability of CLTL [11],[5] over $\mathbb{R}$ with Region Graphs [1] capturing time behavior of variables. Moreover, the satisfiability of CLTLoc can be reduced to an instance of a SMT (satisfiability Modulo Theory) problem. A decision procedure was then easily devised and implemented (`http://code.google.com/p/zot`), by adopting SMT solvers such as Z3 [18]. CLTLoc has been successfully employed to reduce MITL over continuous semantics [6,8], allowing us to implement the first effective tool solving the satisfiability of MITL (`http://code.google.com/p/qtlsolver`).

In this paper, we prove the equivalence of CLTLoc and TA over timed sequences, and that CLTLoc is expressively complete with respect to an extension of the monadic first-order logic used by Kamp in [16], called Timed Monadic First-Order logic (T-MFO). This result extends the Kamp's equivalence between LTL and MFO to timed models. T-MFO is similar to logic $\mathcal{L}_T$ of [3], but it uses a restriction on atomic formulae and suitable *time behavior functions* to represent clocks of CLTLoc. As a consequence, as it is the case for LTL, CLTLoc with past modalities is equivalent to CLTLoc with only future operators. In analogy to TA, the number of clocks that are allowed in CLTLoc formulae determines the expressiveness of the language. We prove, in fact, that there is an infinite hierarchy of languages based on the number of clocks. An arithmetical "next" modality may also be allowed, e.g., to state formulae such as $\mathrm{X}x = y$, meaning that the value of clock x at the next position and the current value of clock y are equal. This modality allows the expression of properties of the length of intervals between two positions, e.g., timed non-regular behaviors where the period can be any real number, which cannot be defined by any temporal logic over dense time so far investigated. The paper is organized as follows: Sect. 2 introduces CLToc, MTL, MITL and TA; Sect. 3 shows the equivalence of CLTLoc and TA; Sect 4 defines the logic T-MFO. showing its equivalence with CLTLoc. Sect. 5 investigates CLTLoc extended with the next arithmetical modality.

2 Languages

Constraint LTL over clocks [7] (CLTLoc) is a semantic fragment of CLTL [11] where formulae are defined with respect to a finite set AP of atomic propositions, a finite set V of clocks and a pair $\mathcal{D} = (\mathbb{R}, \{<, =\})$.

Temporal terms α are defined by the syntax $\alpha := c \mid x \mid \mathrm{X}\alpha$, where c is a constant in $\mathbb{N}$ and $x \in V$. Operator X only applies to temporal terms, with the meaning that $\mathrm{X}\alpha$ is the *value* of temporal term α in the next position.

Formulae are defined as follows:

$$\phi := p \mid \alpha \sim \alpha \mid \phi \wedge \phi \mid \neg\phi \mid \mathbf{X}\phi \mid \mathbf{Y}\phi \mid \phi\mathbf{U}\phi \mid \phi\mathbf{S}\phi$$

where $\sim$ is a relation of $\{<, =\}$ and $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{U}$ and $\mathbf{S}$ are the usual "next", "previous", "until" and "since" operators of LTL, with the same meaning.

For $n \in \mathbb{N}, n \geq 0$, let CLTLoc($\mathrm{X}^n$) denote the class of CLTLoc formulae allowing atomic formulae of the form $\mathrm{X}^h x \sim y$, $\mathrm{X}^h x \sim c$, where x, y are clocks, c is constant and h is an integer with $0 \leq h \leq n$ and let $\mathrm{CLTLoc}^{\mathrm{X}}$ denote the

class of formulae defined as $\bigcup_{n\in\mathbb{N}} \mathrm{CLTLoc}(\mathrm{X}^n)$. Throughout the paper, we write CLTLoc instead of $\mathrm{CLTLoc}(\mathrm{X}^0)$. For convenience, one can still use in CLTLoc the operator X in formulae of the form $\mathrm{X}x \sim c$, as a shorthand for $\mathbf{X}\,(x \sim c)$. Sections 3 and 4 study the expressiveness of CLTLoc, while Section 5 deals with $\mathrm{CLTLoc}^{\mathrm{X}}$. The semantic definitions of $\mathrm{CLTLoc}^{\mathrm{X}}$ in the remainder of this section also apply to its syntactic fragment CLTLoc.

The semantics of $\mathrm{CLTLoc}^{\mathrm{X}}$ is defined with respect to a strict linear order $(\mathbb{N}, <)$ representing positions in time.

The valuation of clocks can be defined by a mapping $\sigma : \mathbb{N}\times V \to \mathbb{R}$, assigning, for every position $i \in \mathbb{N}$, a value $\sigma(i, x)$ to each clock $x \in V$. Intuitively, a clock x measures the time elapsed since the last time when $x = 0$, i.e., the last "reset" of x. To ensure that time progresses at the same rate for every clock, σ is called a *valuation* when satisfies the following condition: for every position $i \in \mathbb{N}$, there exists a "time delay" $\delta_i > 0$ such that for every clock $x \in V$:

$$\sigma(i+1, x) = \begin{cases} \sigma(i, x) + \delta_i, & \text{time progress} \\ 0 & \text{reset } x. \end{cases}$$

By definition of the sequence of δ_i, it follows that time progress is strongly monotonic and, moreover, resets in a valuation are represented by value 0, leading to a very simple definition of $\mathrm{CLTLoc}^{\mathrm{X}}$: there is no distinction between the action of resetting a clock x and of testing whether $x = 0$ in the current valuation. To consider monotonic time progress instead, i.e., allowing $\delta_i \geq 0$, $\mathrm{CLTLoc}^{\mathrm{X}}$ must be enriched with a special operator to represent clock resets whose semantics is different from the one of tests $x \sim c$.

We assume that at every position there is at least one clock which is not reset: if this is not the case, just add a new clock *Now*, which is never reset. Hence, the time delay δ_i is uniquely defined in each position $i > 0$ as $\sigma(i + 1, \mathit{Now}) - \sigma(i, \mathit{Now})$. The initial value of clocks, $\sigma(0, x)$ may be any non-negative value. When comparing CLTLoc with MTL and TA, other assumptions may be introduced to deal with some specific cases, e.g., requiring that clocks start from 0 at position 0 (which is obtained by syntactically imposing $x = 0$ at 0).

An interpretation for a $\mathrm{CLTLoc}^{\mathrm{X}}$ formula ϕ is a pair (π, σ), where σ is a valuation and $\pi : \mathbb{N} \to \wp(AP)$ maps every position to a set of atomic propositions. The semantics of ϕ at position $i \in \mathbb{N}$ over (π, σ) is defined in Table 1. The only case requiring an explanation is the clause for interpreting $\alpha_1 \sim \alpha_2$. Given a temporal term α containing an occurrence of a variable x_α let the *depth* $|\alpha|$ of α be the total amount of temporal shift needed in evaluating α: $|x_\alpha| = 0$ and $|\mathrm{X}\alpha| = |\alpha| + 1$. The value $\sigma(i, \alpha)$ is then defined as: $\sigma(i, \alpha) = \sigma(i + |\alpha|, x_\alpha)$. If α has an occurrence of a constant c_α then $\sigma(i, \alpha) = c_\alpha$. Hence, we can always assume that X does not appear in front of a constant. A $\mathrm{CLTLoc}^{\mathrm{X}}$ formula ϕ is *satisfiable* if $(\pi, \sigma), 0 \models \phi$, for some (π, σ); in this case, (π, σ) is called a *model* of ϕ, and we write $(\pi, \sigma) \models \phi$.

To compare the expressiveness of $\mathrm{CLTLoc}^{\mathrm{X}}$ with other formalisms, we introduce the satisfiability of $\mathrm{CLTLoc}^{\mathrm{X}}$ formulae over timed ω-words (or timed ω-sequences). A timed ω-word over $\wp(AP)$ is a pair (π, τ) where $\pi : \mathbb{N} \to \wp(AP)$

$$
\begin{aligned}
(\pi,\sigma), i \models p &\Leftrightarrow p \in \pi(i) \text{ for } p \in AP \\
(\pi,\sigma), i \models \alpha_1 \sim \alpha_2 &\Leftrightarrow \sigma(i+|\alpha_1|, x_{\alpha_1}) \sim \sigma(i+|\alpha_2|, x_{\alpha_2}) \\
(\pi,\sigma), i \models \mathbf{X}(\phi) &\Leftrightarrow (\pi,\sigma), i+1 \models \phi \\
(\pi,\sigma), i \models \mathbf{Y}(\phi) &\Leftrightarrow (\pi,\sigma), i-1 \models \phi \wedge i > 0 \\
(\pi,\sigma), i \models \phi \mathbf{U} \psi &\Leftrightarrow \exists\, j \geq i : (\pi,\sigma), j \models \psi \ \wedge\ \forall\, i \leq n < j, (\pi,\sigma), n \models \phi \\
(\pi,\sigma), i \models \phi \mathbf{S} \psi &\Leftrightarrow \exists\, 0 \leq j \leq i : (\pi,\sigma), j \models \psi \ \wedge\ j < n \leq i, (\pi,\sigma), n \models \phi
\end{aligned}
$$

Table 1. Semantics of CLTLoc$^{\mathrm{X}}$ (propositional connectives are omitted).

and τ is a monotonic function $\tau : \mathbb{N} \to \mathbb{R}$ such that $\forall i\ \tau(i) < \tau(i+1)$ (strong monotonicity). Without loss of generality, to simplify some of the proofs that follow, we depart slightly from the standard definition of timed words, considering the first position 0 as "special". Given a CLTLoc$^{\mathrm{X}}$ interpretation (π', σ), let τ and π be such that: $\tau(0) = 0$, $\pi(0) = \emptyset$ and for every $i \geq 0$

$$\tau(i+1) = \sigma(i, \mathit{Now}), \pi(i+1) = \pi'(i).$$

Then, (π, τ) is called the timed ω-word associated with (π', σ) and it is denoted by $[(\pi', \sigma)]$. A relation $\models$ can be defined for every timed ω-word (π, τ) and CLTLoc$^{\mathrm{X}}$ formula ϕ as follows. Let $(\pi, \tau) \models \phi$ hold if *there exists* an interpretation (π', σ) such that $(\pi', \sigma) \models \phi$ and $(\pi, \tau) = [(\pi', \sigma)]$. A CLTLoc$^{\mathrm{X}}$ formula ϕ is satisfiable *over timed ω-words*, if $(\pi, \tau) \models \phi$, for some (π, τ).

3 Equivalence of CLTLoc and Timed Automata

This section shows that the set of timed ω-words satisfying a CLTLoc formula is timed ω-regular, i.e., it is accepted by a Timed Automaton. In particular, CLTLoc is an extension of LTL capturing exactly timed ω-regular languages.

We recall the basic definitions of Timed Automata. Let X be a finite set of clocks with values in $\mathbb{R}$. $\Gamma(X)$ is the set of clock constraints over X of the form $x \sim c \mid \neg\gamma \mid \gamma \wedge \gamma$, where $\sim \in \{<, =\}$, $x \in X$ and $c \in \mathbb{N}$. A clock valuation is a function $v : X \to \mathbb{R}$. We write $v \models \gamma$ when the clock valuation satisfies γ. For $t \in \mathbb{R}$, $v + t$ denotes the clock valuation mapping clock x to value $v(x) + t$, i.e., $(v+t)(x) = v(x) + t$.

A Timed Automaton is a tuple $\mathcal{A} = (\Sigma, Q, T, q_0, B)$ where Σ is a finite alphabet, Q is a finite set of control states, $q_0 \in Q$ is the initial state, $B \subseteq Q$ is a subset of control states (corresponding to a Büchi condition) and $T \subseteq Q \times Q \times \Gamma(X) \times \Sigma \times 2^X$ is a set of transitions.

A transition has the form $q \xrightarrow{\gamma, a, S} q'$ where $q, q' \in Q$, γ is a clock constraint of $\Gamma(X)$, a is a symbol of Σ, and S is a set of clocks to be reset. Two transitions $q \xrightarrow{\gamma, a, S} q'$ and $p \xrightarrow{\gamma', b, P} p'$ of T are *consecutive* when $q' = p$. A (finite or infinite) sequence of consecutive transitions in T is a *path* in $\mathcal{A}$. A pair (q, v), where $q \in Q$ and $v : X \to \mathbb{R}$ is a clock valuation, is a *configuration* of $\mathcal{A}$. A *run* ρ of $\mathcal{A}$ over a timed ω-word $(\pi, \tau) \in (\Sigma \times \mathbb{R})^\omega$ is an infinite sequence of configurations

$(q_0, v_0) \xrightarrow[\tau(1)]{\pi(1)} (q_1, v_1) \xrightarrow[\tau(2)]{\pi(2)} \ldots$, where $q_0 \in I$, $v_0(x) = 0$ for all $x \in X$, $v_i(x)$ is either 0 or $v_{i-1}(x) + \tau(i) - \tau(i-1)$ for all $x \in X$ and $i > 0$; moreover, the sequence $q_0 \xrightarrow{\gamma_1, \pi(1), S_1} q_1 \xrightarrow{\gamma_2, \pi(2), S_2} q_2 \ldots$, such that $v_{i-1} + \tau(i) - \tau(i-1) \models \gamma_i$ and $x \in S_i$ iff $v_i(x) = 0$, must be a path of $\mathcal{A}$. Let $inf(\rho)$ be the set of control states $q \in Q$ such that $q = q_i$ for infinitely many positions $i \geq 0$ in ρ. A run is *accepting* when a Büchi condition holds, i.e., $inf(\rho) \cap B \neq \emptyset$.

Since we consider strictly monotonic time sequences τ, a transition with a guard $x = 0$ can never be taken (hence, it can be replaced with *false*): such a transition would be fired at i only when a transition at $i - 1$ resets x, thus entailing $\tau(i) = \tau(i-1)$, contradicting strict monotonicity. From now on, we assume that guards of the form $x = 0$ are not allowed in a TA.

3.1 From Timed Automata to CLTLoc

Following a rather standard approach, we provide a CLTLoc formula which captures the semantics of a TA. To this end, we introduce a set of fresh clocks X_Q representing the control states of $\mathcal{A}$. More precisely, a clock $c_q \in X_Q$ is associated with each control state $q \in Q$; the value of c_q is 0 whenever $\mathcal{A}$ is in q, and it is left to grow (i.e., $c_q > 0$) otherwise. Since in CLTLoc, unlike in TA, a clock cannot be read and reset at the same time, following the approach of [7] for each $x \in X$ we introduce two CLTLoc clocks, x_1 and x_2, which are alternately reset. In addition, we introduce a third clock, x_{12}, which is used to keep track of whether $x_1 < x_2$ (x_{12} is 0 if, and only if, $x_1 < x_2$). We define a set of formulae whose conjunction $\phi_{\mathcal{A}}$ describes a given TA $\mathcal{A}$. The first formula (where $\mathbf{G}(\phi) = \neg(true \mathbf{U} \neg\phi)$) is globally quantified and states that if x_1 is reset then it cannot be reset again, unless x_2 is reset before it; in addition, $x_{12} = 0$ until x_2 is reset:

$$x_1 = 0 \Rightarrow \neg\mathbf{X}\left((x_2 > 0)\mathbf{U}(x_1 = 0)\right) \wedge x_{12} = 0 \ \wedge \mathbf{X}\begin{pmatrix}\mathbf{G}\,(x_{12} = 0 \wedge x_2 > 0) \ \vee \\ (x_{12} = 0)\mathbf{U}(x_2 = 0)\end{pmatrix}.$$

A symmetrical formula is defined also for x_2, but evaluated at position 1 rather than at the origin. A clock constraint $x \sim c$ for a clock x of $\mathcal{A}$ is expressed by the formula $(x_{12} = 0 \wedge x_1 \sim c) \vee (x_{12} > 0 \wedge x_2 \sim c)$. For all $q \in Q$, the following formula translates the transition relation of $\mathcal{A}$:

$$\mathbf{G}(c_q = 0 \Rightarrow \bigvee_{q \xrightarrow{\gamma, a, S} q' \in T} \mathbf{X}\,(a \wedge c_{q'} = 0 \wedge \phi_\gamma \wedge \phi_S)) \tag{1}$$

where ϕ_γ is the CLTLoc formula that captures the clock constraint γ and ϕ_S is the conjunction of formulae of the form $x_1 = 0 \vee x_2 = 0$ for each $x \in S$. The first transition from the initial state must be dealt with separately, because at that position all clocks in the TA are set to 0, by the following formula, to be evaluated at the initial position:

$$\bigvee_{q_0 \xrightarrow{\gamma, a, S} q' \in T} a \wedge c_{q'} = 0 \wedge \phi_\gamma \wedge \phi_S. \tag{2}$$

To represent valid runs of $\mathcal{A}$, suitable formulae are introduced that guarantee the uniqueness of the control state and of the input symbol at each position i. The Büchi acceptance condition is obtained by enforcing that at least one final control state $q_j \in B$ is visited infinitely often. These formulae are rather trivial and are not shown here for brevity. Finally, let $\phi_{\mathcal{A}}$ be the conjunction of all previous CLTLoc formulae that capture the semantics of $\mathcal{A}$.

Theorem 1. *Let $\mathcal{A}$ be a TA with $k \geq 1$ clocks and $n \geq 1$ control states, and let (π, τ) be a timed word over alphabet Σ. Then, (π, τ) is accepted by $\mathcal{A}$ if, and only if, $(\pi, \tau) \models \phi_{\mathcal{A}}$. Moreover, ϕ_A has $3k + n$ clocks.*

3.2 From CLTLoc to Timed Automata

The timed automaton recognizing the language of timed words that are models of a given CLTLoc formula is easily obtained by exploiting the Vardi-Wolper construction [23] for LTL formulae. We take care of clock constraints that are handled as atomic formulae and, in particular, all formulae of the form $x = 0$ are converted into resets. Observe that, with the assumption of strictly monotonic time sequence, CLTLoc formulae $x = 0$ are equivalent to resets of TA.

Theorem 2. *Let ϕ be a CLTLoc formula with k clocks. Then, there exists a k-clock TA $\mathcal{A}_\phi$ recognizing the timed language defined by ϕ: for all timed words (π, τ) over alphabet $\wp(AP)$, $(\pi, \tau) \models \phi$ if, and only if, (π, τ) is recognized by $\mathcal{A}_\phi$.*

From the equivalence of CLTLoc and TA some results follow immediately. The first statement derives from the universality problem for TA.

Corollary 1. *The validity problem for CLTLoc is Π^1_1 complete.*

Let $\text{CLTLoc}_{\mathbf{X},\mathbf{U}}$ be the set CLTLoc formulae with no past operators $\mathbf{S}$ and $\mathbf{Y}$.

Corollary 2. *CLTLoc is equivalent to $\text{CLTLoc}_{\mathbf{X},\mathbf{U}}$.*

The number of clocks plays a crucial role for the expressiveness of TA. In fact, timed regular languages can be arranged in a strict hierarchy, determined by the minimum number of clocks necessary for accepting a given language.

Theorem 3. *[14] For all $k \geq 0$, the class of timed languages accepted by TA with k clocks is strictly included in the class of timed languages accepted by TA with $k + 1$ clocks.*

Example 1. Consider the family of timed languages $\{L_k\}_{k>0}$ where L_k is the set of timed words over the alphabet $\{a\}$ of the form (π, τ), such that $\pi : \mathbb{N} \to \{\emptyset, \{a\}\}$, $\pi(i) = \{a\}$ for all $i > 0$ and there exist at least k distinct pairs (i, j), $0 < i < j \in \mathbb{N}$, such that $\tau(j) - \tau(i) = 1$ (i.e., there are at least k distinct pairs of a's at exactly distance 1). In [22], it was shown that every L_k is accepted by a TA with k clocks, but it cannot be accepted by any TA with $k - 1$ clocks.

The above hierarchy result can easily be extended to CLTLoc. Let CLTLoc^k be the set of CLTLoc formulae where at most k clocks occur, i.e., $|V| \leq k$.

Theorem 4. *For all integers $k > 0$, CLTLoc^k is strictly more expressive than CLTLoc^{k-1}.*

4 Timed Monadic First Order Logic of Orders

We define the logic T-MFO, an extension of the monadic first order logic of order (MFO) that Kamp [16] showed to be equivalent to LTL. We then prove that CLTLoc is expressively complete with respect to T-MFO. T-MFO has two kinds of elements: monadic predicates whose domain is $\mathbb{N}$, and monotonic unary functions $\mathbb{N} \to \mathbb{R}$ relating positions in $\mathbb{N}$ to timestamps in $\mathbb{R}$. Similarly to the logic $\mathcal{L}_T$ of [3], T-MFO includes a special function, denoted as $t : \mathbb{N} \to \mathbb{R}$, associating each discrete position with its absolute timestamp. For simplicity and without loss of generality, in this section we assume that, given a CLTLoc formula, all clocks that appear in it are reset in position 0.

$(\mathbb{N}, <)$ is the theory of discrete positions, whereas $(\mathbb{R}, <, =, +)$ is the structure where timestamps are evaluated. The elements of T-MFO are:

- a set AP of monadic predicates over the set $\mathbb{N}$ of discrete positions;
- relation $<$ and function $+1$ on discrete positions;
- a set T of unary functions $\mathbb{N} \to \mathbb{R}$ from discrete positions to timestamps; one of them is called t;
- relations $<, =$ and function $+1$ on timestamps.

In $(\mathbb{N}, <)$, the constant 0 and the successor function $+1$ can be defined by means of $<$ and first-order quantification, but we introduce them as primitive to overcome the syntactic restrictions introduced next. Let t_x and t_y be unary functions of T from discrete positions to timestamps. We restrict the atomic formulae on timestamps to be of the form $t_x(i) \sim t_y(j) + c$, where $\sim \in \{<, =\}$ and $+c$ ($c \in \mathbb{N}$) corresponds to the application c times of function $+1$. In addition, we impose that atomic formulae only have one free variable. As a consequence, the atomic formulae that can be written on timestamps have the form

$$\beta(i) := t_x(i + h) \sim t_y(i + k) + c$$

where either t_x or t_y may be t and h, k are constants in $\mathbb{N}$ and $\sim \in \{<, =\}$ (the case where both are t is straightforward, because β reduces to either true or false based on $\sim$, h and k).

Function t of T captures the passing of time, and is similar to function f used in [3]. Hence, the following constraint holds: $\forall i \; t(i + 1) > t(i)$. The functions of set T are intended to capture the timestamps when clocks are reset – more precisely, $t_x(i)$ is the last timestamp where clock x is reset. As a consequence, the functions obey the following constraints: $t_x(0) = t(0)$ and

$$\forall i \left((t_x(i + 1) = t_x(i) \vee t_x(i + 1) = t(i + 1)) \right).$$

Finally, formulae of T-MFO are defined by the following grammar (where $p \in AP$ and i, j are variables over $\mathbb{N}$):

$$\phi := p(i) \mid \beta(i) \mid i < j \mid \neg\phi \mid \phi \wedge \phi \mid \forall i \phi.$$

We consider only formulae of T-MFO that do not contain free individual variables. The semantics for T-MFO formulae is defined with respect to a structure

$\mathcal{M}^{\mathcal{I}} = (\mathbb{N}, <, \mathcal{I})$ (or simply $\mathcal{I}$) where interpretation $\mathcal{I}$ specifies the sets $p^{\mathcal{I}} \subseteq \mathbb{N}$ for each $p \in \Sigma$ and the behavior of all functions of T. The satisfaction relation $\models$ is defined in the standard way. A T-MFO formula ϕ is *satisfiable* if there is $\mathcal{I}$ such that $\mathcal{I}, 0 \models \psi$; in this case, we say that $\mathcal{I}$ is a *model* of ϕ.

Given an interpretation $\mathcal{I}$, define the corresponding timed word (π, τ) as:

- For all $p \in AP$, $p \in \pi(i)$ iff $i \in p^{\mathcal{I}}$.
- For all $i \in \mathbb{N}$, $\tau(i) = t(i)$.

Relation $\models$ can be extended to timed words. Let (π, τ) be a timed word and ϕ be a T-MSO formula. We write $(\pi, \tau) \models \phi$ if there exists an interpretation $\mathcal{I}$ that is a model for ϕ such that (π, τ) is obtained from $\mathcal{I}$.

4.1 From CLTLoc to T-MFO

Every CLTLoc formula ϕ can be translated into a T-MFO formula by introducing a monadic predicate $p(i)$ for each CLTLoc proposition p, and a function $t_x(i)$ for each clock x. The following definition of translation r, mapping CLTLoc formulae to T-MFO formulae, follows [3] and is defined inductively on the structure of the CLTLoc formula. First, we introduce mapping r_i, for $i \geq 0$, which is the same as F_i of [3] for $p \in AP$, $\neg$, $\wedge$, $\mathbf{X}$ and $\mathbf{U}$, plus the following:

$$r_i(x \sim c) = t(i) \sim t_x(i) + c.$$

Let ϕ be a CLTLoc formula. Then, $r_0(\phi)$ is the corresponding T-MFO formula.

Theorem 5. *Let ϕ be a CLTLoc formula and (π, σ) be a CLTLoc interpretation. If $(\pi, \sigma) \models \phi$ then there exists an interpretation $\mathcal{I}$ such that $\mathcal{I}, 0 \models r_0(\phi)$. Conversely, let $\mathcal{I}$ be an interpretation such that $\mathcal{I}, 0 \models r_0(\phi)$. Then, there is an interpretation (π, σ) such that $(\pi, \sigma) \models \phi$.*

4.2 From T-MFO to CLTLoc

To obtain the opposite equivalence, we again exploit Kamp's results proving the equivalence between MFO and LTL. To extend the result to T-MFO, we have to show how atomic formulae $\beta(i)$ can be translated into CLTLoc, since MFO does not have formulae in this form.

Theorem 6. *Let ϕ be a T-MFO formula. There exists a CLTLoc formula ϕ' such that, for all timed words (π, τ), $(\pi, \tau) \models \phi$ if, and only if, $(\pi, \tau) \models \phi'$.*

As a consequence of Theorem 6, formulae of the form $t(i) \sim t_y(i + h) + c$ and $t(i + h) \sim t_y(i) + c$ (i.e., where one of t_x, t_y in terms β is t) are enough to characterize timed ω-languages; in fact, by exploiting the equivalence with CLTLoc, one can always remove formulae β where neither t_x nor t_y is t.

Corollary 3. *Let ϕ be a T-MFO formula. Then, there is a T-MFO formula ϕ' without instances of formula $t_x(i+h) \sim t_y(i+k)+c$, with $t_x \neq t \neq t_y$, such that for each timed word (π, τ), it is $(\pi, \tau) \models \phi$ if, and only if, $(\pi, \tau) \models \phi'$.*

5 Timed non-regular languages

Many extensions of the class of TA have been proposed with the goal of increasing its expressiveness. For instance, [1] introduced *diagonal constraints*, i.e., of the form $x \sim y + c$, as guards of transitions. They proved, however, that this extension does not augment the expressiveness of TA, since the construction of the region graph can be generalized to consider diagonal constraints, by refining the equivalence relation $\simeq$ on clock valuations.

In CLTLoc one can also allow diagonal constraints, but as in the case of TA, they do not augment the expressiveness of the language; the occurrence of a formula $x \sim y+c$, with $c \in \mathbb{N}$ a constant in and x, y two clocks, can equivalently be rewritten in CLTLoc as $(x > 0 \wedge y > 0)\mathbf{S}(y = 0 \wedge x \sim c)$. In fact, since time progression is the same for both clocks x and y, the formula $x \sim y+c$ is satisfied at a position $i \in \mathbb{N}$ if there exists a position $j \leq i$ such that at j both $y = 0$ and $x \sim c$ hold, and from position $j+1$ up to i, neither x nor y are reset. Therefore, $x > 0 \wedge y > 0$ holds in every position in the interval $[j+1, i]$.

When considering CLTLoc$^{\mathrm{X}}$, i.e., when temporal terms include also X, the expressive power increases. Notice that atomic formulae of the form $\mathrm{X}^n x \sim \mathrm{X}^m y$, for $m \leq n$, may be ignored since they can equivalently be rewritten as $\mathbf{X}^m(\mathrm{X}^{n-m} x \sim y)$.

Example 2. Let L be the set of timed ω-words over the alphabet $\{a\}$ such that a is periodical. Formally, an ω-word (π, τ) is in L if, and only if, $\pi : \mathbb{N} \to \{\{a\}\}$ and for all $i \in \mathbb{N}$, $\tau(i+2) - \tau(i+1) = \tau(i+1) - \tau(i)$. L is defined in CLTLoc(X) with two clocks x, y as: $y = 0 \wedge x > 0 \wedge \mathbf{G}(a \wedge \mathrm{X}y = x \wedge \mathrm{X}x = y)$. Condition $\mathrm{X}x = y$ states that for all $i \geq 0$ the value of x at position $i+1$ is the same of y at position i. Similarly for $\mathrm{X}y = x$. The formula imposes that at position 0 $y = 0$ and x may assume any real value $\alpha > 0$. Therefore, for all $i \in \mathbb{N}$, if i is even then $\sigma(i, y) = 0, \sigma(i, x) = \alpha$, else $\sigma(i, y) = \alpha, \sigma(i, x) = 0$. Hence, $\sigma(i+1, \mathit{Now}) - \sigma(i, \mathit{Now}) = \alpha$. It is now obvious that (π, σ) is a model of the formula if, and only if, $[(\pi, \sigma)]$ is a timed ω-word of L.

Theorem 7. *The language of Ex. 2 is not timed regular, but it is in CLTLoc*(X).

The following immediate property of operator X is crucial in showing that the timed non-regular language L of Example 2 is in CLTLoc(X).

For a clock $z \in V$, for a position $m \in \mathbb{N}$, let $R(z, m) = \max\{j \mid 0 \leq j \leq m \wedge \sigma(j, z) = 0\}$, i.e., the largest position between 0 and m where z is reset.

Statement 1 *Let $\sigma : \mathbb{N} \times V \to \mathbb{R}$ be a valuation such that all clocks are reset at least at position 0. For $i > 0, x, y \in V$, let $R_x = R(x, i+n)$, and $R_y = R(y, i)$. Then, $\sigma(i, \mathrm{X}^n x) = \sigma(i, y)$ if, and only if, $R_y < R_x$ and:*

$$\sigma(R_x, \mathit{Now}) - \sigma(R_y, \mathit{Now}) = \sigma(i+n, \mathit{Now}) - \sigma(i, \mathit{Now})$$

In fact, $\sigma(i, \mathrm{X}^n x) = \sigma(i+n, x) = \sigma(i+n, \mathit{Now}) - \sigma(R_x, \mathit{Now})$, and $\sigma(y, i) = \sigma(i, \mathit{Now}) - \sigma(R_y, \mathit{Now})$: it follows that $\sigma(i, \mathrm{X}^n x) = \sigma(i, y) \Leftrightarrow \sigma(i+n, \mathit{Now}) -$

$\sigma(R_x, Now) = \sigma(i, Now) - \sigma(R_y, Now)$. Hence, a formula of the form $\mathrm{X}^n x = y$ compares the time distance of positions i and $i+n$ with the time distance of positions R_y and R_x (where y and x were last reset). A special case is when $R_x = i+n$ and $R_y = i$, which entails that $\sigma(i, \mathrm{X}^n x) = \sigma(y, i) = 0$.

For all $k \geq 0$, let $\mathrm{CLTLoc}(\mathrm{X}^n, k)$ be the class of $\mathrm{CLTLoc}(\mathrm{X}^n)$ formulae with at most k clocks. The number of clocks induces an infinite hierarchy over $\mathrm{CLTLoc}(\mathrm{X}^n, k)$:

Theorem 8. *For all $k, n \geq 1$, the class of languages in CLTLoc$(\mathrm{X}^n, k-1)$ is strictly included in the class of languages in CLTLoc(X^n, k).*

We notice that $\mathrm{CLTLoc}(\mathrm{X}^n, k) \not\subseteq \mathrm{CLTLoc}(k+1)$ for $k \geq 2$, since the example of Theorem 7 can be defined with just two clocks, and the language L_k in the proof of Theorem 8 is the same of Ex. 1, which obviously is in CLTLoc^k. Therefore, operator X cannot be used to replace some of the clocks:

Corollary 4. *For all $k \geq 2$, the class of languages in CLTLoc$(k+1)$ is incomparable with the class of languages in CLTLoc(X^n, k).*

The operator X^n also induces an infinite hierarchy:

Theorem 9. *For all $k, n \geq 1$, the class of languages in CLTLoc(X^{n-1}, k) is strictly included in the class of languages in CLTLoc(X^n, k).*

6 Conclusions

This paper studies the expressiveness of Constraint LTL over clocks (CLTLoc), whose main interest of CLTLoc is that its decidability procedure, based on SMT solvers, has actually been implemented, allowing the verification of real time logics such as MITL or QTL.

CLTLoc is equivalent to Timed Automata in the pointwise semantics and it is expressively complete with respect to an extension of Kamp's monadic first-order logic. Its family of languages is organized in an infinite hierarchy based on the number of clocks. When an arithmetical "next" operator is allowed, CLTLoc defines also timed non-ω-regular languages, while still being decidable.

References

1. Alur, R., Dill, D.L.: A theory of timed automata. Theoretical Computer Science 126(2), 183–235 (1994)
2. Alur, R., Feder, T., Henzinger, T.A.: The benefits of relaxing punctuality. Journal of the ACM 43(1), 116–146 (1996)
3. Alur, R., Henzinger, T.A.: Real-time logics: complexity and expressiveness. Information and Computation 104, 390–401 (1993)
4. Alur, R., Henzinger, T.A.: A really temporal logic. Journal of the ACM 41(1), 181–204 (1994)

5. Bersani, M.M., Frigeri, A., Rossi, M., San Pietro, P.: Completeness of the bounded satisfiability problem for constraint LTL. In: Reachability Problems, LNCS, vol. 6945, pp. 58–71 (2011)
6. Bersani, M.M., Rossi, M., San Pietro, P.: On the satisfiability of metric temporal logics over the reals. In: Proc. of the Int. Work. on Automated Verification of Critical Systems (AVOCS). pp. 1–15 (2013)
7. Bersani, M.M., Rossi, M., San Pietro, P.: A tool for deciding the satisfiability of continuous-time metric temporal logic. In: Proc. of TIME. pp. 99–106 (2013)
8. Bersani, M.M., Rossi, M., San Pietro, P.: Deciding continuous-time metric temporal logic with counting modalities. In: Reachability Problems, LNCS, vol. 8169, pp. 70–82 (2013)
9. Bersani, M.M., Rossi, M., San Pietro, P.: Deciding the satisfiability of MITL specifications. In: Proc. of the Int. Symp. on Games, Automata, Logics and Formal Verification (GandALF). pp. 64–78 (2013)
10. Bouyer, P., Markey, N., Ouaknine, J., Worrell, J.: On expressiveness and complexity in real-time model checking. In: Proc. of ICALP. pp. 124–135 (2008)
11. Demri, S., D'Souza, D.: An automata-theoretic approach to constraint LTL. Information and Computation 205(3), 380–415 (2007)
12. Harel, E., Lichtenstein, O., Pnueli, A.: Explicit clock temporal logic. In: LICS. pp. 402–413. IEEE Computer Society (1990)
13. Henzinger, T.A., Raskin, J.F., Schobbens, P.Y.: The regular real-time languages. In: In Proc. 25th Int. Coll. Automata, Languages, and Programming (ICALP'98. pp. 580–591. Springer-Verlag (1998)
14. Henzinger, T.A., Kopke, P.W., Wong-Toi, H.: The expressive power of clocks. In: Proc. of ICALP. pp. 335–346. Springer-Verlag (1995)
15. Hirshfeld, Y., Rabinovich, A.: Quantitative temporal logic. In: Computer Science Logic, LNCS, vol. 1683, pp. 172–187 (1999)
16. Kamp, J.A.W.: Tense Logic and the Theory of Linear Order. Ph.D. thesis, University of California at Los Angeles (1968)
17. Koymans, R.: Specifying real-time properties with metric temporal logic. Real-Time Systems 2(4), 255–299 (1990)
18. Microsoft Research: Z3: An efficient SMT solver. z3.codeplex.com
19. Ostroff, J.S.: Temporal Logic for Real Time Systems. John Wiley & Sons, Inc., New York, NY, USA (1989)
20. Ouaknine, J., Worrell, J.: Safety metric temporal logic is fully decidable. In: In Proceedings of TACAS 06, LNCS 3920. pp. 411–425. Springer (2006)
21. Pradella, M., Morzenti, A., San Pietro, P.: Bounded satisfiability checking of metric temporal logic specifications. ACM Transactions on Software Engineering and Methodology (TOSEM) 22(3) (2013)
22. Suman, P.V., Pandya, P.K.: An introduction to timed automata, chap. 4, pp. 111–146. World Scientific (2012)
23. Vardi, M.Y., Wolper, P.: An automata-theoretic approach to automatic program verification. In: Proc. of LICS. pp. 332–344 (1986)
24. Wilke, T.: Specifying timed state sequences in powerful decidable logics and timed automata (extended abstract). In: FTRTFT. pp. 694–715. No. 863 in LNCS (1994)