2013 20th International Symposium on Temporal Representation and Reasoning

A Tool for Deciding the Satisfiability of Continuous-time Metric Temporal Logic

Marcello M. Bersani*, Matteo Rossi*, Pierluigi San Pietro*†
*Dipartimento di Elettronica Informazione e Bioingegneria, Politecnico di Milano, Italy
† CNR IEIIT-MI, Milano, Italy
Email: {marcellomaria.bersani, matteo.rossi, pierluigi.sanpietro}@polimi.it

***Abstract*—Constraint LTL-over-clocks is a variant of CLTL, an extension of linear-time temporal logic allowing atomic assertions in a concrete constraint system. Satisfiability of CLTL-over-clocks is here shown to be decidable by means of a reduction to a decidable SMT (Satisfiability Modulo Theories) problem. The result is a complete Bounded Satisfiability Checking procedure, which has been implemented by using standard SMT solvers. The importance of this technique derives from the possibility of translating various continuous-time metric temporal logics, such as MITL and QTL, into CLTL-over-clocks itself. Although standard decision procedures of these logics do exist, they have never been realized in practice. Suitable translations into CLTL-over-clocks have instead allowed us the development of the first prototype tool for deciding MITL and QTL. The paper also reports preliminary, but encouraging, experiments on some significant examples of MITL and QTL formulae.**

I. Introduction

Constraint-LTL [1], called CLTL, is an extension of linear-time temporal logic allowing atomic assertions in a concrete constraint system. By carefully choosing the constraint system, CLTL may be decidable, as well as expressive and well-suited to define infinite-state systems and their properties.

In this paper, we define a variant of CLTL, called CLTL-over-clocks (CLTL-oc), where arithmetic variables occurring in atomic assertions are evaluated as *clocks*. A clock "measures" the time elapsed since the last time the clock itself was "reset" (i.e., the variable was equal to 0); clocks can also be tested against a constant. By definition, in CLTL-oc each (discrete) instant i is associated with a "time delay" corresponding to the "time elapsed" between i and the next instant $i+1$. This allows mixing of discrete events with continuous-time, a typical situation arising in many computer-controlled applications.

Satisfiability of CLTL-oc is here shown to be decidable by means of a reduction to a decidable SMT (Satisfiability Modulo Theories) problem, resulting in a complete Bounded Satisfiability Checking procedure. Although other automata-based decision procedures would be suitable to show decidability of CLTL-oc (e.g., [1]), the novelty of our reduction is that it can easily be implemented (by using standard SMT solvers). In fact, the paper also reports on a new tool, publicly available, to verify CLTL-oc formulae.

Although CLTL-oc may be used to specify and verify timed systems, a further advantage of our approach is that it is possible to translate various continuous-time metric temporal logics, such as MITL (Metric Interval Temporal Logic) [2] and QTL [3], into CLTL-oc itself. Standard decision procedures of MITL and QTL logics do already exist (e.g., [2], [4], [5]), typically based on Timed Automata [6], but, to the best of our knowledge, they have never been realized in practice. This may suggest that these approaches are not easily implementable although they were the first used to prove the decidability properties of such logics.

In general, the level of support for verification of continuous-time temporal logics is not as well developed as for discrete-time models. Uppaal [7] is the de-facto standard tool for verification of Timed Automata, but it does not support continuous-time temporal logics: not only satisfiability checking is not available in Uppaal, but even the formalization of system properties in temporal logic is not allowed, aside from rather simple invariants and reachability properties. Satisfiability Modulo Theories is a promising but well-consolidated theory, supported by efficient solvers that are able to decide problems of many disciplines. In particular, decidable SMT problems have been already considered in the recent past, for instance to solve reachability [8] and the bounded version of language inclusion [9] for Timed Automata. The idea is to give a direct representation of bounded runs of Timed Automata through an SMT formula, capturing a bounded unrolling of the transition relation. Similarly, also Bounded Model-Checking of Linear Temporal Logic on Timed Automata [10] can be tackled by reducing the problem to an instance of a SMT problem, by using a technique extending the traditional BMC procedure for LTL finite automata [11], but by restricting the set of valid runs to those that are periodic in the values of the clocks. Finite or periodic runs of Timed Automata can then be encoded in SMT formulae with explicit arithmetic. Nonetheless, also this approach has so far failed to produce a concrete decision procedure for logics such as MITL and QTL. This difficulty is caused by the gap of translating formulae into Timed Automata, a step which is avoided by our approach.

Suitable translations into CLTL-oc have instead allowed us the development of the first available tool for deciding both MITL and QTL, hence showing the generality and effectiveness of our approach. Further evidence of this is provided by the fact that we have also been able to provide a translation of the extension of QTL with so-called Pnueli and counting modalities [12], thus providing its first concrete decision procedure.

We also report preliminary, but encouraging, experiments on some significant examples, such as the timed lamp and its properties, in CLTL-oc, MITL and QTL.

The paper is organized as follows: Sect. II defines CLTL-oc, and illustrates it by means of a running example (a

1530-1311/13 $26.00 © 2013 IEEE
DOI 10.1109/TIME.2013.20

timed lamp). Sect. III proves that CLTL-oc is decidable, while Sect. IV outlines the corresponding SMT-based decision procedure. Sect. V briefly describes MITL and QTL, showing also the general idea behind their translation into CLTL-oc. Sect. VI illustrates the tool, showing verification results of CLTL-oc, MITL and QTL formulae. Sect. VII concludes.

II. CONSTRAINT LTL (OVER CLOCKS)

Constraint LTL (CLTL [1], [13]) is the decidable logic that is used in Sect. V to solve the satisfiability problem of various metric temporal logics over continuous time. CLTL formulae are defined with respect to a finite set V of variables and a *constraint system* $\mathcal{D}$, which is a pair $(D, \mathcal{R})$ with D being a specific domain of interpretation for variables and constants and $\mathcal{R}$ being a family of relations on D, such that the set AP of atomic propositions coincides with set $\mathcal{R}_0$ of 0-ary relations. An *atomic constraint* is a term of the form $R(x_1, \ldots, x_n)$, where R is an n-ary relation of $\mathcal{R}$ on domain D and $x_1, \ldots, x_n$ are variables. A *valuation* is a mapping $v : V \to D$, i.e., an assignment of a value in D to each variable. A constraint is *satisfied* by v, written $v \models_{\mathcal{D}} R(x_1, \ldots, x_n)$, if $(v(x_1), \ldots, v(x_n)) \in R$. Given a variable $x \in V$ over domain D, *temporal terms* are defined by the syntax: $\alpha := c \mid x \mid \mathrm{X}\alpha$, where c is a constant in D and x denotes a variable over D. Operator X is very similar to $\mathbf{X}$, but it only applies to temporal terms, with the meaning that $\mathrm{X}\alpha$ is the *value* of temporal term α in the next time instant. Well-formed CLTL formulae are defined as follows:

$$\phi := R(\alpha_1, \ldots, \alpha_n) \mid \phi \wedge \phi \mid \neg\phi \mid \mathbf{X}(\phi) \mid \mathbf{Y}(\phi) \mid \phi \mathbf{U} \phi \mid \phi \mathbf{S} \phi$$

where α_i's are temporal terms, $R \in \mathcal{R}$, $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{U}$ and $\mathbf{S}$ are the usual "next", "previous", "until" and "since" operators of LTL, with the same meaning. The dual operators "release" $\mathbf{R}$, and "trigger" $\mathbf{T}$ may be defined as usual, i.e., $\phi \mathbf{R} \psi$ is $\neg(\neg\phi \mathbf{U} \neg\psi)$ and $\phi \mathbf{T} \psi$ is $\neg(\neg\phi \mathbf{S} \neg\psi)$.

The semantics of CLTL formulae is defined with respect to a strict linear order representing time $(\mathbb{N}, <)$. Truth values of propositions in AP, and values of variables belonging to V are defined by a pair (π, σ) where $\sigma : \mathbb{N} \times V \to D$ is a function which defines the value of variables at each position in $\mathbb{N}$ and $\pi : \mathbb{N} \to \wp(AP)$ is a function associating a subset of the set of propositions with each element of $\mathbb{N}$. The value of terms is defined with respect to σ as follows:

$$\sigma(i, \alpha) = \sigma(i + |\alpha|, x_\alpha)$$

where x_α is the variable in V occurring in term α and $|\alpha|$ is the *depth* of a temporal term, namely the total amount of temporal shift needed in evaluating α: $|x| = 0$ when x is a variable, and $|\mathrm{X}\alpha| = |\alpha| + 1$. The semantics of a CLTL formula ϕ at instant $i \geqslant 0$ over a linear structure (π, σ) is recursively defined as in Table I, A formula $\phi \in$ CLTL is *satisfiable* if there exists a pair (π, σ) such that $(\pi, \sigma), 0 \models \phi$.

In this paper, we consider a variant of CLTL, where arithmetic variables are evaluated as *clocks* and set $\mathcal{R} \backslash \mathcal{R}_0$ is $\{<, =\}$. A clock "measures" the time elapsed since the last time the clock was "reset" (i.e., the variable was equal to 0). By definition, in CLTL-oc each $i \in \mathbb{N}$ is associated with a "time delay" $\delta(i)$, where $\delta(i) > 0$ for all i, which corresponds to the "time elapsed" between i and the next state $i + 1$. More precisely, for all clocks $x \in V$, $\sigma(i + 1, x) = \sigma(i, x) + \delta(i)$, unless it is "reset" (i.e., $\sigma(i + 1, x) = 0$). We show in Sect. III that CLTL-oc is decidable. Before going further, to motivate our approach, we provide an example of a CLTL-oc formula representing a simple yet realistic timed system.

$$\begin{aligned}
(\pi, \sigma), i &\models p \Leftrightarrow p \in \pi(i) \text{ for } p \in AP \\
(\pi, \sigma), i &\models R(\alpha_1, \ldots, \alpha_n) \Leftrightarrow (\sigma(i + |\alpha_1|, x_{\alpha_1}), \ldots, \sigma(i + |\alpha_n|, x_{\alpha_n})) \in R \\
(\pi, \sigma), i &\models \neg\phi \Leftrightarrow (\pi, \sigma), i \not\models \phi \\
(\pi, \sigma), i &\models \phi \wedge \psi \Leftrightarrow (\pi, \sigma), i \models \phi \text{ and } (\pi, \sigma), i \models \psi \\
(\pi, \sigma), i &\models \mathbf{X}(\phi) \Leftrightarrow (\pi, \sigma), i + 1 \models \phi \\
(\pi, \sigma), i &\models \mathbf{Y}(\phi) \Leftrightarrow (\pi, \sigma), i - 1 \models \phi \wedge i > 0 \\
(\pi, \sigma), i &\models \phi \mathbf{U} \psi \Leftrightarrow \exists\, j \geqslant i : (\pi, \sigma), j \models \psi \wedge \\
&\qquad (\pi, \sigma), n \models \phi \ \forall\, i \leqslant n < j \\
(\pi, \sigma), i &\models \phi \mathbf{S} \psi \Leftrightarrow \exists\, 0 \leqslant j \leqslant i : (\pi, \sigma), j \models \psi \wedge \\
&\qquad (\pi, \sigma), n \models \phi \ \forall\, j < n \leqslant i
\end{aligned}$$

TABLE I. SEMANTICS OF CLTL.

Example 1: we consider the LTL specification of a timed lamp and its properties (studied in Sect. VI) from [14]. The lamp is controlled by two buttons, labeled ON and OFF respectively, which cannot be pressed simultaneously. The lamp itself can be either on or off. When ON is pressed the lamp is immediately turned on, regardless of its current state, while if OFF is pushed then the lamp is immediately turned off, also regardless of its current state. After ON is pressed, the lamp will not stay on forever, but, if no more buttons are pressed, it will automatically turn off with a delay Δ, a positive real constant. By pressing the ON button before the timeout expiration then the timeout is extended by a new delay Δ.

Our CLTL-oc formula makes use of atomic propositions *on*, *off* and l representing, respectively, events "push button ON" and "push button OFF" and the state "light is on". Clocks may be used to measure the exact time elapsed since the last *on*; clearly some clock must be "reset" (i.e., set to 0, in analogy to Timed Automata) whenever ON is pressed, while when a clock is equal to Δ then the timeout *expires.*Since the introduction of clocks is not straightforward, we first define a few shorthands called *rst-c*, $\mathit{test}_{c=\Delta}$ and $\mathit{test}_{0<c\leqslant\Delta}$. They have the intuitive meaning (which will be formalized after the main specification) that they are true if, and only if, a clock c is reset or, respectively, $c = \Delta$, or $0 < c \leqslant \Delta$. The specification of the lamp, still lacking the precise clock specification, is defined by the formula $\mathbf{G}\left(\bigwedge_{i=0}^{5}(i)\right)$, where $\mathbf{G}(\phi)$ is the usual globally operator defined by $\bot \mathbf{R} \phi$, of the following formulae:

$$\neg(\mathit{on} \wedge \mathit{off}) \tag{1}$$

$$\mathit{on} \Leftrightarrow \mathit{rst\text{-}c} \tag{2}$$

$$\mathbf{Y}(l) \Rightarrow \mathit{test}_{0<c\leqslant\Delta} \tag{3}$$

$$\mathit{turnoff} \Leftrightarrow \mathbf{Y}(l) \wedge (\mathit{off} \vee \mathit{test}_{c=\Delta}) \tag{4}$$

$$l \Leftrightarrow \neg \mathit{turnoff}\ \mathbf{S}\ \mathit{on}. \tag{5}$$

Formula (1) ensures mutual exclusion; (2) states that the timeout must be (re)started whenever button ON is pressed; (3) constrains the time elapsed since the previous instant if the light was on at that moment (i.e., not more than Δ); (4) defines (for readability) an event *turnoff*, capturing the two cases when the lamp (supposed to be ON in the previous instant) must be turned off at the current instant (i.e., OFF being pressed or

the timeout expiring); finally, (5) gives the specification of the light, as being on if, and only if, there was in the past an *on* event not followed by a *turnoff*. Initialization is implicit in the specification (at instant 0, the light is off).

To complete the specification, we must formalize also the clock behavior. In CLTL-oc, "resetting a clock" c, e.g., following an *on* event is as simple as stating that $on \Rightarrow c = 0$; testing a clock c against a constant Δ and causing say, a *turnoff* is a simple as stating that $c = \Delta \Rightarrow turnoff$. Unfortunately, the same clock cannot be tested and reset at the same time. A possible solution is to introduce two clocks c_0 and c_1, rather than just one clock, so that they are reset alternatively: only one of the two clocks is reset and a new reset of the same clock will eventually occur only after the occurrence of a reset of the other clock. The behavior of this clock pair is described by the axiom $\mathbf{G}\,((6) \wedge (7))$, where formulae (6) and (7) are:

$$\bigwedge_{i\in\{0,1\}} \big(c_i = 0 \Rightarrow \neg\mathbf{X}\,\big((c_{(i+1)_2} > 0\,\mathbf{U}\,c_i = 0)\big)\big) \qquad (6)$$

$$c_0 = 0 \Rightarrow \neg(c_1 = 0) \qquad (7)$$

and $()_2$ stands for the modulo 2 operator (i.e., $(1)_2 = 0$, $(2)_2 = 0$). Finally, the above clock shorthands *rst-c*, $test_{c=\Delta}$ and $test_{0<c\leqslant\Delta}$ are defined as follows:

$$\textit{rst-c} \Leftrightarrow c_0 = 0 \vee c_1 = 0$$

$$test_{0<c\leqslant\Delta} \Leftrightarrow \bigvee_{i\in\{0,1\}} 0 < c_i \leqslant \Delta$$

$$test_{c=\Delta} \Leftrightarrow \bigvee_{i\in\{0,1\}} \big(c_i = \Delta \wedge (c_{(i+1)_2} > \Delta \vee c_{(i+1)_2} = 0)\big)$$

III. Decidability of CLTL over clocks

Our argument is divided in two parts. First, in this section, we show that a CLTL-oc formula ϕ can be translated into a suitable Büchi automaton A^R_ϕ, which recognizes the language of the regions of the clocks appearing in ϕ. If the automaton accepts some ultimately periodic words of the form uv^ω over the alphabet of clock regions induced by ϕ, then ϕ is satisfiable. In fact, through arguments akin to those in [6], we can show that an infinite sequence of successive regions is representative of a trace of a CLTL-oc formula. Automaton A^R_ϕ is built using a slight variation of the construction used in [1] and [13]; it differs from those built in the earlier works in that its symbolic *runs* include the sequences of *symbolic valuations* that satisfy ϕ [1] and also constraints representing the clock regions induced by ϕ. Second, in Sect. IV we provide a way to solve in practice the satisfiability of CLTL-oc, through the method used in [13] to solve satisfiability of CLTL. The technique relies on encoding CLTL formulae into formulae of a decidable fragment of first-order logic, which can then be solved by off-the-shelf SMT solvers. The decision procedure hinges on finding a finite sequence of assignments to the clocks appearing in the CLTL formula ϕ, which satisfies ϕ and such that it is a witness of an ultimately periodic sequence of successive clock regions. Since the clock regions define a partition of the space of clock assignments, each assignment of values to clocks uniquely identifies a region, hence an exhaustive definition of all the regions is not needed.

To represent correctly the elapsing time as measured by clocks, symbolic models of CLTL-oc formulae has to define a sequence of clock regions $R_0R_1,\ldots$ such that, for all time position i, R_{i+1} is a time-successor of region R_i [6], except for the clocks that in R_{i+1} are reset (i.e., whose value is 0). Let ϕ be a CLTL-oc formula and $\mathcal{A}_\phi$ be its Büchi automaton recognizing symbolic models of ϕ [15]. Then, we define an automaton $\mathcal{A}^R_\phi$ accepting all the symbolic models belonging to $\mathscr{L}(\mathcal{A}_\phi)$, such that the sequence of regions entailed by clock constraints within symbolic valuations obeys the time-successor relation. Let $\mathcal{A}^c_R = (\mathcal{R}, \delta)$ be the automaton where $\mathcal{R}$ is the set of all regions induced by the clocks, c is the maximum constant that all clocks are compared with and δ is the transition relation containing all $R \to R'$ such that $R' \in \mathcal{R}$ is a time-successor of $R \in \mathcal{R}$, except for the clocks whose value is 0 in R'. Automaton $\mathcal{A}^R_\phi$ is defined as the product of $\mathcal{A}_\phi$ and automaton $\mathcal{A}^c_R$ recognizing the language of successive regions, which are induced by formula ϕ. It is worthy to be noticed that the following construction only guarantees that time elapses and all clocks progress by the same amount of time. Since we are dealing with a CLTL formulae where atoms may be relations over clocks, the construction of $\mathcal{A}^R_\phi$ does not force the value of clocks which is instead constrained by the formula. This impacts the definition of the initial region, which does not follow the standard construction where all clocks are zero. In our case, we relax the assumption that all clocks start from 0, by simply considering each region as potentially initial.

We briefly recall some key elements of $\mathcal{A}_\phi$ (see [13] for further details). Let $SV(\phi)$ denote the set of symbolic valuations associated with ϕ. The closure of ϕ, denoted $cl(\phi)$, is the smallest set containing all subformulae of ϕ and closed under negation. An *atom* $\Gamma \subseteq cl(\phi)$ is a maximally consistent subset of formulae of $cl(\phi)$, i.e., such that, for each subformula ξ in ϕ, either $\xi \in \Gamma$ or $\neg\xi \in \Gamma$. Automaton A_ϕ is the tuple $(SV(\phi), Q, Q_0, \eta, F)$ where Q is the set of all the atoms and η is defined as in the standard Vardi-Wolper construction.

Now, we define A^R_ϕ as tuple $(SV(\phi), Q \times \mathcal{R}, \mathcal{I}, \eta', F \times \mathcal{R})$. Relation η' is defined as follows: $(\Gamma, R) \xrightarrow{sv\cup R} (\Gamma', R')$ if, and only if, $sv \in SV(\phi)$ satisfies Γ, the pair (Γ, Γ') is one-step consistent, $R \to R' \in \delta$ and $sv \cup R$ is satisfiable. Set $\mathcal{I} \subseteq Q_0 \times \mathcal{R}$ of initial states consists of states (atoms) of Q_0 that are consistent with respect to regions in R; i.e., the set of arithmetical constraints in $\Gamma \in Q_0$ union R is a satisfiable conjunction of formulae.

A run ρ of A^R_ϕ is a sequence:

$$(\Gamma_0, R_0) \xrightarrow{sv_0\cup R_0} \ldots (\Gamma_i, R_i) \xrightarrow{sv_i\cup R_i} (\Gamma_{i+1}, R_{i+1}) \ldots$$

where $(\Gamma_0, R_0) \in \mathcal{I}$. Let (π, σ) be a sequence where $\pi : \mathbb{N} \to \wp(AP)$ and $\sigma : \mathbb{N} \times V \to \mathbb{R}$. (π, σ) *witnesses* ρ, i.e., $(\pi, \sigma) \models \rho$, when for all $i \geqslant 0$:

$$\pi(i) = \Gamma_i \cap AP \text{ and } \sigma, i \models sv_i \cup R_i.$$

Lemma 1. *Let ϕ be a CLTL-oc formula. $(\pi, \sigma) \models \phi$ if, and only if, there is an accepting run ρ of $\mathcal{A}^R_\phi$ such that $(\pi, \sigma) \models \rho$.*

Proof: We first show that, if $(\pi, \sigma) \models \phi$, then there is an accepting run of automaton $\mathcal{A}^R_\phi$. In [1] it is shown that a CLTL formula ϕ is satisfiable if, and only if, there is a run $\rho = sv_0sv_1\ldots$ of automaton $\mathcal{A}_\phi$ recognizing symbolic models of the formula. We have to show that when the variables of ϕ behave like clocks, then the sequence of valuations defines a

sequence of successive clock regions, hence ρ is also a run of $\mathcal{A}_\phi^R$. In fact, we have that $\sigma, i \models sv_i$. Since the set of clock regions is a partition of the clocks space, there is only one region $R \in \mathcal{R}$ such that $\sigma, i \models R_i$, with $R_i = R$. Therefore, $\sigma, i \models sv_i \cup R_i$ and model (π, σ) induces a sequence of regions $R_0 R_1 \ldots$. By construction, each atom Γ_i is such that $\pi(i) = \Gamma_i \cap AP$. Finally, we prove that each $R_i R_{i+1}$ in the sequence $R_0 R_1 \ldots$ is a pair of successive regions. In fact, if $\sigma(i, x)$ and $\sigma(i+1, x)$ are two adjacent valuations for a clock x of ϕ, then either there is a $t > 0$ such that $x(i+1) = x(i) + t$ or $x(i+1) = 0$ (reset). Therefore, R_{i+1} is a time successor of R_i (except for the clocks whose value is 0 in R_{i+1}) and the sequence $R_0 R_1 \ldots$ is a sequence of successive regions which belongs to the language $\mathscr{L}(\mathcal{A}_R^c)$. At each position i, we have that $\pi(i) = \Gamma_i \cap AP$ and $\sigma(i) \models sv_i \cup R_i$. Hence, $(\pi, \sigma) \models \rho$.

We now show that, if automaton $\mathcal{A}_\phi^R$ has an accepting run ρ, then CLTL formula ϕ is satisfiable. The Vardi-Wolper construction of the Büchi automaton $\mathcal{A}_\phi$ guarantees that at each position of the run ρ a subset of the subformulae of ϕ are satisfied and ϕ is witnessed at position 0. By construction, the sequence of symbolic valuation $sv_0 sv_1 \ldots$ is a symbolic model of ϕ and such that it admits arithmetical model; i.e., there is an infinite sequence of valuations of the variables occurring in ϕ and satisfying the constraints in sv_i, for all i [1]. By construction, the sequence $R_0 R_1 \ldots$ is a sequence of successive clock regions in the language $\mathscr{L}(A_R^c)$. Therefore, by standard arguments from Timed Automata, we can build a sequence of delays $\delta(i)$ such that, for each clock $x \in V$, it is $x(i+1) = x(i) + \delta(i)$, unless x is reset (i.e. $x(i) = 0$). ■

Observe that time progression in automaton $\mathcal{A}_\phi^R$ is not guaranteed by the construction. However, this requirement is easily achieved by the CLTL formula: $\mathbf{G}(\mathrm{X}x = 0 \vee \mathrm{X}x > x) \wedge (\mathbf{GF}(x = 0) \vee \mathbf{FG}(x > c(x)))$ (where $c(x)$ is the biggest constant clock x is compared to), for all clocks $x \in V$.

Finally, the following result is a direct consequence of Lemma 1 and of properties of CLTL and Timed Automata.

Theorem 1. *Satisfiability for CLTL-oc is decidable.*

Proof: Let ϕ be a CLTL-oc formula. By Lemma 1 we can build automaton $\mathcal{A}_\phi^R$, recognizing symbolic models of ϕ, that has an accepting run (i.e., whose accepted language is non-empty) iff ϕ is satisfiable. Checking the emptiness of the language $\mathscr{L}(\mathcal{A}_\phi)$ is done by standard techniques which look for cycles in the graph of $\mathcal{A}_\phi^R$ embedding (at least) one accepting control state. The language of $\mathcal{A}_\phi^R$ is not empty if there exists a path of $\mathcal{A}_\phi^R$ of the form

$$(\Gamma_0, R_0) \ldots (\Gamma_{l-1}, R_{l-1})(\Gamma_l, R_l) \ldots (\Gamma_k, R_k)$$

where $\Gamma_k = \Gamma_l$, $R_k = R_l$ and all atoms belonging to F occur at least once in $(\Gamma_{l-1}, R_{l-1})(\Gamma_l, R_l) \ldots (\Gamma_{k-1}, R_{k-1})$. The word which is recognized by the run is an ultimately periodic one over the language $SV(\phi) \cup \mathcal{R}$ of the form:

$$(sv_0, R_0) \ldots (sv_{l-1}, R_{l-1}) \left((sv_l, R_l) \ldots (sv_{k-1}, R_{k-1})\right)^\omega .$$

■

1) Complexity.: The satisfiability problem for CLTL-oc is PSPACE-hard, as any LTL formula (whose satisfiability problem is PSPACE-complete) is also a CLTL formula. We can show the PSPACE-completeness for CLTL-oc using arguments similar to those used in [6] to show how the transition relation of the automaton checked for language emptiness can be computed in PSPACE. In fact, consider a CLTL-oc formula ϕ. If we indicate with $|\phi|$ the number of subformulae of ϕ, with N the number of clock variables in ϕ, and with K the biggest constant against which the clock variables of ϕ are compared, since the number of clock regions is $O(N! \cdot K^N)$ [6], the number of states of automaton $\mathcal{A}_\phi^R$ is $O(2^{|\phi|} \cdot N! \cdot K^N)$. However, to check for the emptiness of $\mathscr{L}(\mathcal{A}_\phi^R)$ we do not need to build the whole state space, but only a constant number of vertices at a time. Since the space needed to store a vertex of automaton $\mathcal{A}_\phi^R$, when using a binary encoding for K, is polynomial in $|\phi| \log(K)$, the algorithm for checking the emptiness of $\mathcal{A}_\phi^R$ is in PSPACE.

IV. Solving CLTL-over-clocks satisfiability

We outline how to decide the satisfiability problem for CLTL-oc formulae by a SMT-based technique instead of automata. The technique is based on previous work [16] [15], in which we used a k-bounded satisfiability problem to solve the satisfiability of CLTL formulae by using a polynomial reduction to a SMT problem. k-bounded satisfiability is complete, and deciding the satisfiability of a CLTL formula can be done by means of a finite amount of k-bounded satisfiability tests, for increasing values of k. To deal with variables that behave like clocks, the method developed in [15] is extended to represent time progress.

Given a CLTL formula ϕ, we say that ϕ is k-bounded satisfiable if there exists an ultimately periodic sequence of symbolic valuations $sv_0, \ldots, sv_{l-1}(sv_l \ldots sv_{k-1})^\omega$, which is a symbolic model of ϕ and such that there is a partial assignment of values to all the variables occurring in ϕ only for a finite number of positions in time, from 0 to $k+1$. In other words, in k-bounded satisfiability we look for a finite sequence of symbolic valuations $sv_0, \ldots, sv_{l-1}(sv_l \ldots sv_{k-1} sv_k)$, where $sv_k = sv_l$, which admits a k-bounded arithmetical model and that is representative of an infinite symbolic model for ϕ of the form $sv_0, \ldots, sv_{l-1}(sv_l \ldots sv_{k-1})^\omega$. While in PSPACE, in practice k-bounded satisfiability can be quite efficient, at least when the value of k is small enough to perform the check: checking k-bounded satisfiability is then equivalent to solve a few SMT problems in P. Obviously, the upper bound for k is in general exponential in the size of the formula.

In [15] we show how to solve k-bounded satisfiability for CLTL formulae over a class of arithmetical constraints that include the family of clock constraints used in Sect. II. The k-bounded satisfiability problem is solved through a polynomial-time reduction to the satisfibility problem of a formula in the theory of Equality and Uninterpreted Functions combined with Linear Integers/Reals Arithmetic. The combination of the two theories is decidable and its decision procedure is implemented by many SMT-solvers. The reduction of [15] has been implemented in the `ae2zot` plugin of the Zot tool [17]. Therefore, an instance of the k-bounded satisfiability problem for CLTL formulae has the complexity of the underlying SMT problem, which depends on the arithmetic theory required. In our case, since clocks are in $\mathbb{R}$, we solve SMT problems in QF-EUF $\cup$ LRA, whose complexity is P. A peculiarity of the SMT-based approach is that, if the set of symbolic

valuations partitions of the space $D^{|V|}$ (with D the domain of the variables in V), then a sequence of valuations uniquely induces a sequence of symbolic valuations. By solving the k-bounded satisfiability problem for a formula ϕ we obtain, from the model of the QF-EUF formula, the sequence of valuations satisfying the constraints occurring in the formula, which induces a symbolic model. Hence, unlike automata-based techniques, our approach does not require the explicit construction of the set $SV(\phi)$. We exploit this to avoid building the set of clock regions induced by CLTL formulae.

To solve the satisfiability of CLTL-oc we still use k-bounded satisfiability to look for ultimately periodic models, but we extend the method in order to represent clock regions and time progression. Representing clock regions is quite straightforward and exploits the fact that regions partition the space of all possible clock valuations. In other words, a clock valuation identifies a clock region, so it is not necessary to precompute the set of all clock regions from the formula. The only requirement to be enforced is the periodicity of the sequence of clock regions corresponding to clock valuations. If one is looking for a model of length k, the sequence of clock regions is of the form: $R_0 \dots R_{l-1}(R_l \dots R_{k-1})^\omega$, which is obtained from a finite sequence $R_0 \dots R_{l-1}(R_l \dots R_{k-1}R_k)$ with the periodicity constraint $R_l = R_k$. The QF-EUF encoding of CLTL formulae is defined to enforce periodicity of all atomic formulae (atomic propositions and clock constraints) between positions k and l. For instance, given two clocks x, y, if $x = y$ holds at position k (i.e., $x(k) = y(k)$) then, by the periodicity constraints, it must also hold at position l: $x(k) = y(k) \Leftrightarrow x(l) = y(l)$. To obtain a periodic sequence of regions we provide the solver with all the clock constraints which may occur in the definition of regions in $\mathcal{R}_\phi$ (but not all the regions). Given a CLTL-oc formula ϕ, we define the set $ac(\mathcal{R}_\phi)$ as the set of all clock constraints induced by ϕ. Let ϕ' be a CLTL formula, $C_{\phi'}$ be the set of clocks appearing in ϕ', and $x, y \in C_{\phi'}$. If $c(x)$ is the maximum constant with which clock x is compared in ϕ' and $\sim \in \{<, =, >\}$, the set $ac(x) = \{x \sim c \mid \forall c \in [0, 1, \dots, c(x)]\} \backslash \{x < 0\}$ is the set of all clock constraints between x and constant $c(x)$, while set $ac(x, y) = \{x \sim y\}$ is the set of all clock constraints between x and y. Then, set $ac(\mathcal{R}_\phi)$ is defined as: $\bigcup_{x,y \in C_{\phi'}, x \neq y} ac(x) \cup ac(x, y)$. We indicate with $Per(ac(\mathcal{R}_\phi))$ the set of QF-EUF periodicity constraints on arithmetic constraints $ac(\mathcal{R})$, that is, $\forall \theta \in ac(\mathcal{R})\ \theta(k) \Leftrightarrow \theta(l)$. Time elapsing is obtained by forcing all clocks in ϕ' to progress by the same amount of time between two positions of the model. Let $i \in [0, k-1]$; then, constraint $x(i+1) = x(i) + \delta(i) \vee x(i+1) = 0$, where $\delta(i) > 0$, is imposed to represent a positive elapsing time, for all clocks $x \in C_{\phi'}$. We indicate with $Adv(C_{\phi'})$ the set of constraints that impose the uniform time advancement of clocks in $C_{\phi'}$. Since all variables in the theory of Reals are defined over $\mathbb{R}$ and all $x \in C_{\phi'}$ behave like clocks, we force $x(i) \geqslant 0$ for all $x \in C_{\phi'}$ and for all $i \in [0, k]$. We indicate the conjunction of these constraints as $nNeg(C_{\phi'})$.

We solve the satisfiability problem of CLTL-oc formula ϕ by feeding the SMT solver the set of constraints $|\phi'|_k \cup Per(ac(\mathcal{R}_\phi)) \cup Adv(C_{\phi'}) \cup nNeg(C_{\phi'})$, where $|\phi'|_k$ is the bounded representation of ϕ' described in [15].

The complete definition of $|\phi|_k$ can be found in [15].

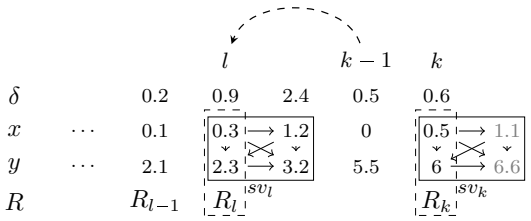

Fig. 1. A (portion of) bounded model satisfying infinitely often formula $(y > 0)\mathbf{U}(x = 0)$. $Per(ac(\mathcal{R}_\phi))$ enforces regions R_l and R_k (dashed rectangles) to be equal, where $R_l = \{x_l < y_l, 1 < x_l < 2, 2 < y_l\}$ and $R_k = \{x_k < y_k, 1 < x_k < 2, 2 < y_k\}$. Solid rectangles represent symbolic valuations $sv_l = sv_k$ where constraints $x \sim c$ are not written because included in the regions. The set of all constraints $a \to b$ (representing $a < b$) between all the pairs of elements in $\{x, y, 0, 2\}$ defines a symbolic valuation. Value $x_{k-1} = 0$ satisfies constraint $\mathbf{G}(\mathbf{F}(x = 0))$ and values y_i, with $i \in [l-1, k-1]$, witnesses $\mathbf{F}(\mathbf{G}(y > c(y)))$. The complete definition of $|\phi|_k$ can be found in [15]. We provide some of its formulae for the sake of space. Formula $1 \leqslant l \leqslant k-1$ sets position l of the loop in $[1, k-1]$. The semantics of the formula $\phi = (y > 0)\mathbf{U}(x = 0)$ is achieved by means of the fixpoint definition of $\mathbf{U}$ over all positions in $0, k$ that is $\bigwedge_{i=0}^{k} \phi_i \Leftrightarrow (x_i = 0) \vee (y_i > 0 \wedge \phi_{i+1})$. The semantic equivalence of position k and l, i.e., all the subformulae of ϕ have the same truth value at k and l, is enforced by $\phi_l \Leftrightarrow \phi_k, (x_l = 0) \Leftrightarrow (x_k = 0), (y_l > 0) \Leftrightarrow (y_k > 0)$. The eventuality of $(x = 0)$ in the periodic part, when ϕ holds at position k, is guaranteed by $\phi_k \Rightarrow l \leqslant i_\phi \leqslant k \wedge (x_{i_\phi} = 0)$.

V. Encoding Metric Temporal Logics

We exploit the decision procedure for CLTL-oc outlined in Sect. IV to define mechanisms for deciding various metric temporal logics over continuous time. In fact, we have defined several satisfiability-preserving reductions from metric temporal logics to CLTL-oc; hence, satisfiability of formulae of these former logics can be determined by solving the corresponding problem for CLTL-oc. In particular, the logics we have targeted so far are: MITL [2], QTL [18], and QTL with counting and Pnueli modalities [12]. In this section, after briefly recalling the definition of MITL and QTL, we present some highlights from the aforementioned reductions.

Let AP be a finite set of atomic propositions. The syntax of (well formed) formulae of MITL is defined as follows:

$$\phi := p \mid \phi \wedge \phi \mid \neg\phi \mid \phi \mathbf{U}_I \phi$$

with $p \in AP$ and I is an interval of the form $\langle a, b \rangle$ or of the form $\langle a, +\infty \rangle$, where $a, b \in \mathbb{N}$ are constants, with $a < b$. Interval $\langle a, b \rangle$ can be, in general, left open or closed (respectively, (or [) but only right closed (written as]). We omit the interval I when it is $[0, +\infty)$.

$$\begin{aligned}
M, t &\models p \Leftrightarrow p \in M(t) \qquad p \in AP \\
M, t &\models \neg\phi \Leftrightarrow M, t \not\models \phi \\
M, t &\models \phi \wedge \psi \Leftrightarrow M, t \models \phi \text{ and } M, t \models \psi \\
M, t &\models \phi \mathbf{U}_I \psi \Leftrightarrow \exists t' \in t + I : M, t' \models \psi \text{ and } M, t'' \models \phi\ \forall t'' \in [t, t')
\end{aligned}$$

TABLE II. Semantics of MITL.

The semantics of MITL is defined in Table II with respect to *signals*. A signal is a function $M : \mathbb{R}_+ \to 2^{AP}$ (with $\mathbb{R}_+$ the set of nonnegative reals). A MITL formula ϕ is *satisfiable* if there exists a signal M such that $M, 0 \models \phi$ (in this case, M is called a *model* of ϕ). The *globally* $\mathbf{G}_I$ and *eventually* $\mathbf{F}_I$

operators can be defined by the usual abbreviations: $\mathbf{F}_I\phi = \top\mathbf{U}_I\phi$ and $\mathbf{G}_I\phi = \neg\mathbf{F}_I(\neg\phi)$.

QTL is similar to MITL, but it is based on a single metric operator, $\mathbf{F}_{(0,1)}$ (and its past counterpart $\mathbf{P}_{(0,1)}$), so its syntax is the following:[1]

$$\phi := p \mid \phi \wedge \phi \mid \neg\phi \mid \phi\mathbf{U}_{(0,\infty)}\phi \mid \mathbf{F}_{(0,1)}\phi \mid \phi\mathbf{S}_{(0,\infty)}\phi \mid \mathbf{P}_{(0,1)}\phi.$$

For brevity, we omit the semantics of QTL, which is however similar to the one in Tab. II. Note that, despite its apparent simplicity, QTL has the same expressive power as MITL [19].

We now briefly show how to encode MITL and QTL formulae into CLTL-oc ones . For space reasons we only provide some highlights of the reduction in a special case.

Let us restrict signals to those where intervals are left-closed and right-open (l.c.r.o. signals, ···●——○ ●——○···). In addition, given a MITL formula ϕ we are interested in signals that are non-Zeno [20] models of ϕ, i.e., such that in every finite interval of the temporal domain there is a finite number of change points of the value of the atomic propositions of ϕ. For these (l.c.r.o.) signals, the temporal domain can be partitioned in a countable set of adjacent l.c.r.o. intervals such that in each such interval the value of every subformula of ϕ is constant. Then, for each subformula θ of ϕ ($\theta \in subf(\phi)$) we introduce a CLTL-oc predicate $\overleftarrow{\theta}$ that represents the value of θ in the intervals in which the temporal domain is partitioned. We also introduce the following abbreviations:

$$\underset{\leftarrow}{\xi} = \neg\overleftarrow{\xi} \qquad \lrcorner_\xi = \neg\mathbf{Y}(\overleftarrow{\xi}) \wedge \overleftarrow{\xi} \qquad \llcorner_\xi = \neg\mathbf{Y}(\underset{\leftarrow}{\xi}) \wedge \underset{\leftarrow}{\xi}$$

where $\lrcorner_\xi$, for example, captures the situation in which ξ changes value from false to true, with the formula being true in the current interval.

For simplicity, we focus our attention on temporal operators $\mathbf{F}_{(0,b]}(\psi)$ and $\mathbf{P}_{[0,b)}(\psi)$. We remark that it can be shown that, if ψ holds only in l.c.r.o. intervals, so do $\mathbf{F}_{(0,b]}(\psi)$ and $\mathbf{P}_{[0,b)}(\psi)$ (the same does not hold, for example, for $\mathbf{F}_{(0,b)}(\psi)$). For each subformula $\theta \in subf(\phi)$ we introduce two clocks, z^0_θ and z^1_θ, which measure the time from the last change point (either $\lrcorner_\theta$ or $\llcorner_\theta$, so we have $\lrcorner_\theta \vee \llcorner_\theta \Leftrightarrow z^0_\theta = 0 \vee z^1_\theta = 0$), and whose resets alternate.

Formula (8), then, captures the condition in which formula $\theta = \mathbf{F}_{(0,b]}(\psi)$ becomes false: in this case, ψ must become false, and it cannot become true again for b instants (i.e., ψ cannot become true again before its associated clock that is reset when ψ becomes false hits b).

$$\llcorner_\theta \Leftrightarrow \llcorner_\psi \wedge \lrcorner_\psi \mathbf{R} \neg \left(\lrcorner_\psi \wedge \bigwedge_{i\in\{0,1\}} z^i_\psi \leqslant b \right) \tag{8}$$

The case for θ becoming true is not shown for brevity.

Consider the case $\theta = \mathbf{P}_{[0,b)}(\psi)$. Formula (9) captures the condition in which θ becomes true. This occurs when ψ becomes true and either the current instant is the origin (O is an abbreviation for $\neg\mathbf{Y}(\top)$), or ψ has never become true since the origin, or the last time ψ changed value (necessarily from false to true), this occurred more than b instants ago (i.e., the clock associated with ψ that is not reset now is $\geqslant b$).

$$\lrcorner_\theta \Leftrightarrow \lrcorner_\psi \wedge \left(O \vee \mathbf{Y}\left(\neg\lrcorner_\psi \mathbf{S} \left(O \wedge \underset{\leftarrow}{\psi} \right) \right) \vee \bigvee_{i\in\{0,1\}} z^i_\psi \geqslant b \right) \tag{9}$$

To conclude this section, we provide an example of MITL formula (to be evaluated over l.c.r.o. signals) over two atomic propositions p, q whose model is intrinsically aperiodic in the values of the delays between changepoints in the values of p and q. The existence of formulae admitting only aperiodic models shows that, in the decision procedure of Sect. IV, the periodicity must be enforced on the set of constraints defining regions, but not on the actual values of the clocks, nor on the time differences δ; i.e., the encoding of CLTL formula ϕ' does not include constraints $\delta(k) = \delta(l)$ and $x(k) = x(l)$. In other words, for the example below there does not exist a periodic sequence of time increments $\delta(0)\delta(1)\ldots(\delta_l \ldots \delta_{k-1})^\omega$ representing the time elapsing for aperiodic models even if the sequence of clock regions is periodic.

Example 2: consider the conjunction of the following MITL formulae defining the behaviour of two signals p, q as depicted in Figure 2. Signal p holds in $[2k, 2k+1+\varepsilon)$, for all k, and is false elsewhere, as formalized by the MITL formulae

$$\mathbf{G}_{\langle 0,1]}p \wedge \mathbf{G}(\mathbf{G}_{\langle 0,1]}p \Rightarrow \mathbf{G}_{\langle 2,3]}p) \tag{10}$$

$$\mathbf{G}_{\langle 0,1]}q \wedge \mathbf{G}(\mathbf{G}_{\langle 0,1]}q \Rightarrow \mathbf{G}_{\langle 2,3]}q). \tag{11}$$

Both signals p and q hold over intervals longer than one time unit, because of the l.c.r.o. assumption. In addition, we require that q is at least as long as p by means of formula $\mathbf{G}\,(p \Rightarrow q)$. Formulae (10)-(11) admit, in general, models that can be periodic; therefore, we have to restrict the set only to aperiodic models. This may be achieved by enforcing that, over intervals of the form $[2k+1, 2k+2]$, with $k \geqslant 0$, signal q is strictly longer than p, while over intervals $[2k+1.5, 2k+2)$ both p and q are false, as required by the followig two formulae:

$$\mathbf{G}(\mathbf{G}_{\langle 0,1]}(p \wedge q) \Rightarrow \mathbf{F}_{\langle 1,2]}((\neg p \wedge q)\mathbf{U}(\mathbf{G}_{\langle 0,0.5]}\neg q)) \tag{12}$$

$$\mathbf{G}(\neg p \wedge q \Rightarrow \mathbf{G}_{\langle 1,2]}p) \tag{13}$$

Predicates p and q become false before each position $2k+1.5$ by requiring that $\neg p \wedge q$ occurs until an interval of length 0.5 where both p and q are false. Let $t^p_k \in (2k+1, 2k+1.5)$ be the instant where p becomes false, and $t^q_k = t^p_k + \delta_k \in (2k+1, 2k+1.5)$ the one where q becomes false; let $\delta_k = t^p_k - t^q_k$ be the length of the interval where p does not occur while q does. Formula (13) lengthens signal p of δ_k time units over the next interval starting from $2k$: p holds in $[t^p_k + 1, t^p_k + 2 + \delta_k)$. The series of values δ_k is strictly monotonic decreasing because each value is arbitrarily strictly less than the previous one, i.e., $\delta_k > \delta_{k+1}$, for all k. Therefore, the sequence of $\delta(i)$ is not periodic, although the sequence of clock regions of induced by the clocks in the CLTL-oc formula corresponding to the formulae above is indeed periodic.

VI. Implementation & Experimental Results

The decision procedure of Sect. IV for CLTL-oc is implemented in a plugin, called `ae2zot`, of our Zot toolkit [17], whereas the reductions outlined in Sect. V are implemented in the `qtlsolver` tool, available from [21]. The tool translates

[1] Note that MITL can be enriched with the "metric since" $\mathbf{S}_I$ past operator.

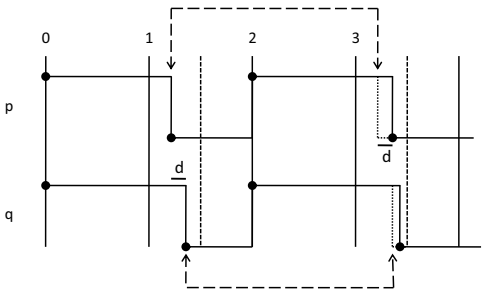

Fig. 2. Aperiodic model for MITL formula of Example 2

MITL or QTL into CLTL-oc, which can be checked for satisfiability by `ae2zot`.

The resulting toolkit has a 3-layered structure, where CLTL-oc is the intermediate layer between SMT-solvers and various temporal formalisms that can be reduced to CLTL-oc. This not only supports (bounded) satisfiability verification of different languages, but it also allows the expression of different degrees of abstraction. For instance, QTL and MITL abstract away the notion of clocks, inherently encompassed within temporal modalities, which are instead explicit in CLTL-oc (as witnessed by the example of the timed lamp in Sect. II) and available to a user, e.g., to express or verify properties where clocks are very convenient. In fact, preliminary experimental results point out that the time required to solve CLTL-oc may be significantly smaller than the one needed for more abstract classes of languages, such as MITL. This gap is caused by the "effort" required to capture the semantics of temporal modalities, which, on the other hand, allow for more concise and manageable high-level specifications. One can then take advantage of the layered structure, which allows the resolution of a formula to be compliant also with constraints imposed at lower layers, for instance by adding at the CLTL-oc layer some extra formula limiting the set of valid models (e.g., by discarding certain edges of some events or by adding particular timing requirements). Also the third layer (the SMT solver) may be used to add further constraints, e.g., to force the occurrence of a proposition or of a certain clock value at a specific discrete position of the finite model.

The current implementation of `qtlsolver` supports various reductions. More precisely, it realizes the MITL-to-CLTL-over-clock translation tailored to l.c.r.o. signals, as highlighted in Sect. V. It also implements a translation from a generalized version of QTL to CLTL-oc. This translation does not assume any special shape for signals, except that they be finitely variable; it natively supports operators $\mathbf{F}_{\langle 0,b\rangle}$ and $\mathbf{G}_{\langle 0,b\rangle}$ (and their past counterparts), where the bounds can be either included or excluded. These operators allow us to define concisely $\mathbf{F}_{\langle a,b\rangle}$ and $\mathbf{G}_{\langle a,b\rangle}$ as abbreviations. For instance, $\mathbf{G}_{(3,6)}(\phi)$ is equivalent to $\mathbf{G}_{(0,3)}\big(\mathbf{F}_{(0,3)}\big(\mathbf{G}_{(0,3)}(\phi)\big)\big)$; defining a similar equivalence using only the $\mathbf{F}_{(0,1)}$ and $\mathbf{G}_{(0,1)}$ modalities (see, e.g., [3]) involves the recursive expansions of each conjunct of $\mathbf{G}_{(3,4)}(\phi) \wedge \mathbf{G}_{[4,5)}(\phi) \wedge \mathbf{G}_{[5,6)}(\phi)$, where $\mathbf{G}_{(n,n+1)}(\phi)$ is equivalent to $\mathbf{G}_{(n-1,n)}\big(\mathbf{F}_{(0,1)}\big(\mathbf{G}_{(0,1)}(\phi)\big)\big)$. The following two encodings are currently available:

MITL providing a direct definition of MITL operators, assuming l.c.r.o. intervals;

QTL providing the definition of generalized QTL operators with unrestricted signals (other than they be finitely variable), and MITL operators through abbreviations.

We used the above two encodings and the CLTL-oc decision procedure to carry out some verification experiments on the example of the Timed Lamp described in Sect. II. More precisely, we have built several descriptions of the behavior of the lamp: (i) the CLTL-oc model presented in Sect. II; (ii) a MITL specification assuming l.c.r.o. signals; (iii) a QTL specification in which predicates *on* and *off* are constrained to be true only in isolated instants. On each of these specifications we have carried out three experiments, assuming $\Delta = 5$: a check of the satisfiability of the specification, to show that it is consistent (*sat*); the (dis)proof of property "the light never stays on for more than Δ time units" (p_1); the proof of property "if at some point the light stays on for more than Δ time units, then there is an instant when *on* is pressed, and then it is pressed again before Δ time units" (p_2). Depending on the temporal logic and of the restrictions on the signals (l.c.r.o. or not) the formalization of the timed lamp and of the properties can change.

In the case of the CLTL-oc specification of the timed lamp, in order to formalize properties p_1 and p_2 we introduce an auxiliary clock c_{aux}, which is reset every time the light is turned on, i.e., $c_{\mathrm{aux}} \Leftrightarrow l \wedge \mathbf{Y}(\neg l)$. Then, in CLTL-oc property p_1 is captured by formula $\mathbf{G}(\mathbf{Y}(l) \Rightarrow c_{\mathrm{aux}} \leqslant \Delta)$. In addition, property p_2 is formalized by the following formula:

$$\mathbf{F}(l \wedge c_{\mathrm{aux}} \geqslant \Delta) \Rightarrow \mathbf{F}(\mathit{on} \wedge \mathbf{X}(\neg \mathit{rst}\text{-}c\mathbf{U}(\mathit{on} \wedge \mathit{test}_{0<c\leqslant\Delta}))) \quad (14)$$

The behavior of the timed lamp can be captured by the following MITL formula over l.c.r.o. signals (we write $\mathbf{G}$ for $\mathbf{G}_{[0,\infty)}$, and $\mathbf{S}$ for $\mathbf{S}_{[0,\infty)}$):

$$\mathbf{G}\left(\left(l \Leftrightarrow (\neg \mathit{off}\ \mathbf{S}\ \mathit{on}) \wedge \mathbf{P}_{[0,\Delta)}(\mathit{on})\right) \wedge (\mathit{on} \Rightarrow \neg \mathit{off})\right) \quad (15)$$

In MITL over l.c.r.o. signals, where predicates hold over non-null intervals, we limit the length of intervals in which *on* (and *off*) holds to be at most 1 by adding the following constraint:

$$\mathbf{G}\big(\neg \mathbf{G}_{(0,1]}(\mathit{on}) \wedge \neg \mathbf{G}_{(0,1]}(\mathit{off})\big)\,. \quad (16)$$

Over unrestricted signals, instead, we force *on* to hold only in isolated instants by adding QTL constraint (similarly for *off*)

$$\mathbf{G}\big(\neg(\mathit{on}\ \mathbf{U}_{(0,+\infty)}\top) \wedge \neg(\mathit{on}\ \mathbf{S}_{(0,+\infty)}\top)\big)\,. \quad (17)$$

Properties p_1 and p_2 over unrestricted signals are captured by the following QTL formulae (where $\mathbf{F}$ stands for $\mathbf{F}_{[0,+\infty)}$):

$$\mathbf{G}\big(\mathbf{F}_{[0,\Delta]}(\neg l)\big) \quad (18)$$

$$\mathbf{F}\big(\mathbf{G}_{[0,\Delta]}(l)\big) \Rightarrow \mathbf{F}\big(\mathit{on} \wedge \mathbf{F}_{(0,\Delta]}(\mathit{on})\big) \quad (19)$$

Over l.c.r.o. signals property p_1 is still captured by Formula (18); property p_2, instead, is more involved, and corresponds to the following formula:

$$\mathbf{F}\big(\mathbf{G}_{[0,\Delta]}(l)\big) \Rightarrow \mathbf{F}\big((\neg \mathit{on} \wedge \mathbf{P}_{[0,\Delta)}(\mathit{on}))\mathbf{U}\ \mathit{on}\big) \quad (20)$$

Table VI reports the time and space required for the checks outlined above (all tests have been done using the Common Lisp compiler SBCL 1.1.2 on a 2.13GHz Core2 Duo MacBook Air with MacOS X 10.7 and 4GB of RAM; the solver was z3 4.0). All bounded satisfiability checks have been performed using a bound $k = 20$. The first line of each row shows

TABLE III. EXPERIMENTAL RESULTS WITH THE TIMED LAMP, REPORTING TIME (SEC) AND HEAP SIZE (MB).

Problem	Satisfiable?	CLTL-o-c	MITL (l.c.r.o)	QTL (unrest.)
sat	Yes	0.48/0.33 5.63	15.5/13.84 66.45	4.24/3.04 27.12
$\mathbf{p_1}$	Yes	0.52/0.35 6.22	36.74/33.16 102.47	17.2/14.86 63.5
$\mathbf{p_2}$	No	0.67/0.49 6.55	6.61/5.09 110.27	257.1/240.88 58.66

the total processing time (i.e., parsing and solving) and the time taken by the SMT-solver (both times in seconds). The second line reports the heap size (in Mbytes) required by Z3. In every case the specification is satisfiable, property p_1 does not hold (the tool returns a counterexample), while property p_2 holds ("unsat" is returned). In addition to the results shown in the table, a variant of Formula (14) where $test_{0<c<\Delta}$ is used instead of $test_{0<c\leqslant\Delta}$ (i.e., $\leqslant$ is replaced by $<$) is shown to not hold, and a counterexample is obtained in less than 1 second.

Finally, we present an interesting behavior over unrestricted signals. The behavior is captured by the following formulae, which state that p and q only occur in isolated instants, with p occurring exactly every 80 time units, and q occurring within 80 time units in the past from each p (origin excluded).

$$\mathbf{G}\left(\begin{array}{c}\mathbf{G}_{(0,80)}(\neg p) \Rightarrow \mathbf{G}_{(80,160)}(\neg p) \quad \wedge \\ (p \Rightarrow \mathbf{F}_{(0,160)}p) \ \wedge \ (q \Rightarrow (\neg q)\,\mathbf{U}\,\top)\end{array}\right) \wedge \\ p \ \wedge \ \mathbf{G}_{(0,80)}(\neg p) \ \wedge \ \mathbf{G}_{(0,\infty)}(p \Rightarrow \mathbf{P}_{(0,80)}q) \tag{21}$$

In this case, the bound $k = 10$ is enough to prove that the formula is satisfiable and a model is produced in about 40 secs. In around the same time the solver shows that property $\mathbf{G}(p \Rightarrow \mathbf{F}_{(0,80)}(q))$ holds for model (21) (up to the considered bound), whereas property $\mathbf{G}(q \Rightarrow \mathbf{F}_{(0,80)}(q))$ does not. Note that, in Formula (21), the constants involved in the temporal modalities are significantly greater than the bound k required to obtain a model. In fact, any value is possible in principle for the clock increments between two consecutive discrete instants, controlled by the (nondeterministic) variable δ. This highlights that the length of the intervals described by a CLTL-oc model is independent of the bound k, as long as this is big enough to capture all changepoints that are necessary to build a periodic sequence of regions.

VII. CONCLUSIONS

This paper investigates a bounded approach to satisfiability checking of an extension of CLTL where variables behave like clocks (CLTL-oc). The decidability of the logic (by means of an automata-based technique) is shown first, followed by an encoding into a decidable SMT problem. This encoding, implemented in our `ae2zot` tool, allows, both in principle and in practice, the use of SMT solvers to check the satisfiability of CLTL-oc. We provide a short but non-trivial example of a CLTL-oc specification describing a timed behavior over continuous time, which should demonstrate the effectiveness of this approach, as we are able to (dis)prove various properties of the specification. The paper also outlines the encoding of two continuous time, metric temporal logics, namely MITL and QTL, that are implemented in our `qtlsolver` tool. This shows that CLTL-oc can be considered as a target language to reduce decision problems of various continuous-time formalisms (not limited to logics, but in principle also Timed Automata or Timed Petri Nets).

To the best of our knowledge, our approach is the first allowing an effective implementation of a fully automated verification tool for continuous-time metric temporal logics such as MITL and QTL. The tool is still a non-optimized prototype, whose performance might also be substantially improved in future versions. Still, verification of formulae requiring many clocks may always be infeasible, since satisfiability of MITL is EXPSPACE-complete (but we also support verification also of an interesting, PSPACE-complete fragment of MITL). However, in practice a large number of clocks is not very frequent, and the examples of MITL (and QTL) formulae that we studied were verified in a fairly short time.

REFERENCES

[1] S. Demri and D. D'Souza, "An automata-theoretic approach to constraint LTL," *Inf. Comput.*, vol. 205, no. 3, pp. 380–415, 2007.

[2] R. Alur, T. Feder, and T. A. Henzinger, "The benefits of relaxing punctuality," *Journal of the ACM*, vol. 43, no. 1, pp. 116–146, 1996.

[3] Y. Hirshfeld and A. Rabinovich, "Timer formulas and decidable metric temporal logic," *Inf. and Comp.*, vol. 198, no. 2, pp. 148 – 178, 2005.

[4] O. Maler, D. Nickovic, and A. Pnueli, "From MITL to timed automata," in *Proc. of FORMATS*, ser. LNCS, 2006, vol. 4202, pp. 274–289.

[5] P.-Y. Schobbens, J.-F. Raskin, and T. A. Henzinger, "Axioms for real-time logics," *Theor. Comput. Sci.*, vol. 274, no. 1-2, pp. 151–182, 2002.

[6] R. Alur and D. L. Dill, "A theory of timed automata," *Theoretical Computer Science*, vol. 126, no. 2, pp. 183–235, 1994.

[7] J. Bengtsson and W. Yi, "Timed automata: Semantics, algorithms and tools," in *Lect. on Concurrency and Petri Nets*, ser. LNCS. Springer, 2004, vol. 3098, pp. 87–124.

[8] P. Niebert, M. Mahfoudh, E. Asarin, M. Bozga, O. Maler, and N. Jain, "Verification of timed automata via satisfiability checking," in *FTRTFT*, ser. LNCS, 2002, vol. 2469, pp. 225–243.

[9] B. Badban and M. Lange, "Exact incremental analysis of timed automata with an SMT-solver," in *FORMATS*, ser. LNCS, 2011, vol. 6919, pp. 177–192.

[10] G. Audemard, A. Cimatti, A. Kornilowicz, and R. Sebastiani, "Bounded model checking for timed systems," in *Proc. of FORTE*, 2002, pp. 243–259.

[11] E. M. Clarke, D. Kroening, J. Ouaknine, and O. Strichman, "Completeness and complexity of bounded model checking," in *VMCAI*, ser. LNCS, vol. 2937. Springer, 2004, pp. 85–96.

[12] A. Rabinovich, "Complexity of metric temporal logics with counting and the Pnueli modalities," *Th. Comp. Sci.*, vol. 411, pp. 2331–2342, 2010.

[13] M. M. Bersani, A. Frigeri, M. Rossi, and P. San Pietro, "Completeness of the bounded satisfiability problem for constraint LTL," in *Reachability Problems*, ser. LNCS, 2011, vol. 6945, pp. 58–71.

[14] M. Pradella, A. Morzenti, and P. San Pietro, "Bounded satisfiability checking of metric temporal logic specifications," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 2013, to appear.

[15] M. M. Bersani, A. Frigeri, A. Morzenti, M. Pradella, M. Rossi, and P. San Pietro, "CLTL Satisfiability Checking without Automata," arXiv:1205.0946v1, 2012.

[16] ——, "Bounded reachability for temporal logic over constraint systems," in *TIME*. IEEE Computer Society, 2010, pp. 43–50.

[17] "Zot: a bounded satisfiability checker," available from `zot.googlecode.com`.

[18] Y. Hirshfeld and A. Rabinovich, "Quantitative temporal logic," in *Computer Science Logic*, ser. LNCS, 1999, vol. 1683, pp. 172–187.

[19] Y. Hirshfeld and A. M. Rabinovich, "Logics for real time: Decidability and complexity," *Fund. Inf.*, vol. 62, no. 1, pp. 1–28, 2004.

[20] C. A. Furia, D. Mandrioli, A. Morzenti, and M. Rossi, *Modeling Time in Computing*, ser. EATCS Monographs in Th. C. Sci. Springer, 2012.

[21] "qtlsolver," available from `qtlsolver.googlecode.com`.