Towards a Theory of Time-Bounded Verification

Joël Ouaknine and James Worrell

Oxford University Computing Laboratory, UK
{joel,jbw}@comlab.ox.ac.uk

Abstract. We propose a theory of *time-bounded* verification for real-time systems, in which verification queries are phrased over time intervals of fixed, bounded duration. We argue that this theory is both ***pertinent***, in that it is fully adequate to handle a large proportion of 'real-world' real-time systems and specifications; and ***effective***, in that the restriction to bounded time domains reclaims as decidable several of the key decision problems of unbounded real-time verification. Finally, we discuss several directions of ongoing and future work.

1 Introduction

In an influential invited address at the 10th Annual IEEE Symposium on Logic in Computer Science (LICS 95), Boris Trakhtenbrot urged the research community to *"lift the 'classical' Trinity to real-time systems"* [51]. Trakhtenbrot's 'Trinity' consisted of Logic, Nets, and Automata, viewed as the pillars of the 'classical' theory of verification. The genesis of this theory in fact go back some five decades to the seminal work of Büchi, Elgot, and Trakhtenbrot himself relating the monadic second-order logic of order ($\mathsf{MSO}(<)$) and automata; see [53] for a detailed historical perspective on the subject.

Underlying the increasingly successful applications of verification technology to the design and validation of hardware and software systems has been the long-running and sustained elaboration of a rich body of theoretical work. One of the major accomplishments of this theory is the discovery and formulation of the robust and far-reaching correspondence among the eclectic concepts of *automata*, *temporal logic*, *monadic predicate logic*, and *regular expressions*. Each of these comes in various flavours, yet the adequation is maintained, in particular, whether the discourse is over the finite or the infinite, or (in the language of predicate logic) first or second order. A key result in this area is Kamp's theorem, which asserts the expressive equivalence of the monadic first-order logic of order ($\mathsf{FO}(<)$) and Linear Temporal Logic (LTL) [29, 19]. This influential result has largely contributed to the emergence of LTL as the canonical linear-time specification formalism in the classical theory.

On a pragmatic level, the close relationship between automata and logic has enabled the design of *model-checking* algorithms for a wide variety of specification formalisms rooted in temporal or predicate logic. While initially little more than pure decidability results, these procedures have over the last few decades been progressively honed into powerful industrial-strength tools.

Real-time verification, by contrast, is a much younger field. Its origins date back approximately twenty-five years, when various researchers from such diverse communities as process algebra, Petri nets, automata theory, and software engineering began investigating extensions of existing formalisms to adequately handle timing considerations. By far the most prominent modelling paradigm to have emerged is Alur and Dill's notion of *timed automaton* [2], which at the time of writing has accrued nearly 4000 citations according to Google Scholar.

One of the central results concerning timed automata is the PSPACE decidability of the *language emptiness* (or *reachability*) *problem* [1]. Unfortunately, the *language inclusion problem*—given two timed automata $\mathcal{A}$ and $\mathcal{B}$, is every timed word accepted by $\mathcal{A}$ also accepted by $\mathcal{B}$?—is known to be undecidable [2]. A closely related phenomenon is the fact that timed automata are not closed under complement. For example, the automaton in Fig. 1 accepts every timed word in which there are two a-events separated by exactly one time unit.

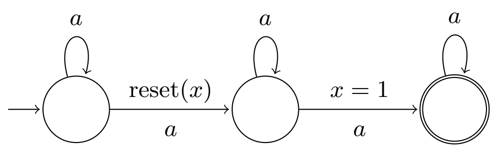

Fig. 1. An uncomplementable timed automaton.

The complement language consists of all timed words in which no two a-events are separated by precisely one time unit. Intuitively, this language is not expressible by a timed automaton, since such an automaton would need an unbounded number of clocks to keep track of the time delay from each a-event. (We refer the reader to [25] for a rigorous treatment of these considerations.)

The undecidability of language inclusion severely restricts the algorithmic analysis of timed automata, both from a practical and theoretical perspective, as many interesting questions can be phrased in terms of language inclusion. Over the past two decades, several researchers have therefore attempted to circumvent this obstacle by investigating language inclusion, or closely related concepts, under various assumptions and restrictions. Among others, we note the use of (i) topological restrictions and digitisation techniques: [22, 14, 42, 39]; (ii) fuzzy semantics: [20, 23, 41, 8]; (iii) determinisable subclasses of timed automata: [4, 47]; (iv) timed simulation relations and homomorphisms: [50, 37, 31]; and (v) restrictions on the number of clocks: [43, 18]. See also Henzinger *et al.*'s paper on *fully decidable* formalisms [24].

In a strictly formal sense, the non-closure under complementation is easy to remedy—one can simply generalise the transition mode to allow both conjunctive and disjunctive transitions, an idea borrowed from the theory of untimed automata that dates back thirty years [15]. Such untimed *alternating automata* have played key roles in algorithms for complementing Büchi automata (see, e.g.,

[33]), temporal logic verification [52, 36], and analysis of parity games [17]. In the timed world, the resulting *alternating timed automata* [34, 44, 35, 46, 16] subsume ordinary timed automata and can be shown to be closed under all Boolean operations. They have been used, among others, to provide model-checking algorithms for various fragments of Metric Temporal Logic (MTL); see, e.g., [44, 45, 12]. Unfortunately, the price to pay for the increase in expressiveness is the undecidability of language emptiness!

Turning to temporal logic, one finds a considerable body of work in the literature concerned with adapting classical linear temporal formalisms to the real-time setting; see, e.g., [32, 38, 7, 3, 54, 48]. One of the earliest proposals, Koyman's *Metric Temporal Logic* (MTL) [32], extends LTL by constraining the temporal operators by (bounded or unbounded) intervals of the reals. For example, the formula $\Diamond_{[3,4]}\,\varphi$ requires φ to hold within 3 to 4 time units from the present time. MTL has since become one of the most successful and popular specification formalisms for timed systems.

Unfortunately, the *satisfiability* and *model-checking problems* for MTL are undecidable [21]. This has led researchers to consider various restrictions on MTL to recover decidability. One of the most important such proposals is *Metric Interval Temporal Logic* (MITL), a fragment of MTL in which the temporal operators may only be constrained by *non-singular* intervals. Alur *et al.* showed that the satisfiability and model-checking problems for MITL are EXPSPACE-complete [3]. A significant extension of this result, based on the notion of *flatness*, was later obtained in [13]. Another interesting approach is that of Wilke, who considered MTL over a dense-time semantics with *bounded variability*, i.e., parameterised by a fixed bound on the number of events per unit time interval [54]. Wilke showed that the satisfiability problem is decidable in this semantics and that MTL with existential quantification over propositions is precisely as expressive as Alur-Dill timed automata.

Work on real-time extensions of monadic first- and second-order logic of order has been considerably scarcer. Hirshfeld and Rabinovich examine the *monadic first-order metric logic of order* ($\mathsf{FO}(<,+1)$) and show that unfortunately, it is—in a precise technical sense—strictly more expressive over the reals than any 'reasonable' timed temporal logic, and in particular than MTL [27]; see also [11]. This sweeping inequivalence seriously dampens the hope of discovering a 'canonical' timed temporal logic over the reals with a natural predicate-logical counterpart, after the manner of LTL and $\mathsf{FO}(<)$ in the classical theory.

There has also been comparatively little research on finding suitable timed analogues of the notion of regular expression. An interesting proposal is that of Asarin *et al.* [9], who define a class of *timed regular expressions* with expressive power precisely that of Alur-Dill timed automata, mirroring Kleene's theorem in the classical theory. Unfortunately, many natural questions, such as whether two timed regular expressions are equivalent, remain undecidable.

In our view, the overall emerging picture of the present-day theory of real-time verification is one of an amalgam of constructs and results—some deep and striking—yet fundamentally constrained by a phalanx of inescapable undecidability barriers. The elegance, uniformity, and canonicity of the classical theory are lacking, and Trakhtenbrot's challenge to a large extent remains unmet.

In an attempt to address these issues, we would like to propose here a ***time-bounded theory of real-time verification***. By 'time-bounded' we mean to restrict the modelling and verification efforts to some bounded interval of time, which itself can be taken as a parameter. A proximate motivation for our proposal is the analogy with *bounded model checking*, which aims to circumvent an intractable verification task by performing instead a more restricted—but less costly—analysis. A related paradigm, originating from economics, is that of *discounting the future*, whereby the later a potential error may occur, the lesser of a concern it is.

Note that while bounded model checking restricts the total number of allowable events (or discrete steps), time-bounded verification restricts the total duration under consideration, but *not* the number of events, which can still be unboundedly large owing to the density of time. We argue that this restriction on total duration is a very natural one as regards real-time systems. For example, a run of a communication protocol might normally be expected to have an *a priori* time bound, even if the total number of messages exchanged is potentially unbounded. In fact, many real-time systems, such as the flight control software on board an aircraft, are normally rebooted and reset at regular intervals (for example, presumably, on completion of a successful flight). One might even argue that most hard real-time problems, which typically involve deadlines, timeouts, and delays, only exist within a finely circumscribed time span. In such cases, a time-bounded analysis seems entirely pertinent. We note that several researchers have in fact already considered instances of time-bounded verification in the context of real-time systems [49, 10, 30].

Aside from these practical considerations, we anticipate more favourable complexity-theoretic properties from a time-bounded theory than from its unbounded counterpart. In recent work [40, 28], we have already amassed considerable evidence to this effect, which we survey below and detail at greater length in the main body of this paper.

The undecidability of language inclusion for timed automata, first established in [2], uses in a crucial way the unboundedness of the time domain. Roughly speaking, this allows one to encode arbitrarily long computations of a Turing machine. In [40], we turned to the *time-bounded* version of the language inclusion problem: given two timed automata $\mathcal{A}$ and $\mathcal{B}$, together with a time bound N, are all finite timed words of duration at most N that are accepted by $\mathcal{A}$ also accepted by $\mathcal{B}$? One of our main results is that this problem is decidable and in fact 2EXPSPACE-complete. It is worth noting that the time-boundedness restriction does not alter the fact that timed automata are not closed under complement, so that classical techniques for language inclusion do not trivially apply.

In subsequent work, we examined the substantially more sophisticated problem of time-bounded emptiness (or equivalently, language inclusion) for alternating timed automata [28]. We also succeeded in establishing decidability, but in contrast to ordinary timed automata, showed that this problem has non-elementary complexity.

A third line of investigation concerns the relative expressiveness of temporal and predicate metric logics over bounded intervals of the reals, in analogy with the classical equivalence of LTL and FO($<$). Somewhat surprisingly, we discovered that MTL has precisely the same expressive power as FO($<, +1$) over any bounded time domain [40]. This is in sharp contrast to the situation over unbounded time, where neither MTL nor any 'reasonable' temporal extension of it can match the full expressiveness of FO($<, +1$) [27].

Finally, we devoted a significant fraction of our efforts to time-bounded model-checking and satisfiability questions for timed automata and metric logics. In addition to MTL and FO($<, +1$), we consider the *monadic second-order metric logic of order*, MSO($<, +1$). In [40], we showed that the time-bounded model-checking and satisfiability problems for monadic first- and second-order metric logics all have non-elementary complexity, whereas these problems are EXPSPACE-complete in the case of MTL (and this in spite of the expressive equivalence of MTL and FO($<, +1$) over bounded time domains). It is worth recalling, in contrast, that these problems are all undecidable over unbounded time.

We believe that this small but significant body of results constitutes a clear indication that the restriction to time-boundedness may lead to a substantially better-behaved theory of real-time verification, mirroring the classical theory and enabling one to lift classical results to the timed world.

It is perhaps worth stressing that we do not envisage time-bounded verification to replace its unbounded counterpart entirely; one can always imagine instances genuinely requiring unbounded real-time analysis. What we do assert, however, is that for a large proportion of hard real-time systems, a time-bounded approach should prove not only algorithmically advantageous, but will also be entirely adequate theoretically.

The remainder of the paper is organised as follows. We recall standard real-time definitions and conventions in Sec. 2. Sections 3, 4, and 5 respectively introduce ordinary timed automata, metric logics, and alternating timed automata. In Sec. 6, we turn to the relative expressiveness of MTL and FO($<, +1$) over bounded time domains. Section 7 then examines our various time-bounded decision problems: emptiness, language inclusion, model checking, and satisfiability. Finally, we briefly discuss some of the multiple possible future research directions in Sec. 8.

Our treatment is fairly spare; in particular, we do not present proofs, but instead offer pointers to the relevant literature. Our aim is mainly to motivate and illustrate, and we have occasionally opted to sacrifice precision for insight.

2 Real-Time Preliminaries

We fix some of the real-time notation and modelling conventions that we use throughout this paper. While there are a wealth of alternatives and variants that can be considered—many of which appear in some form or other in the literature—our aim here is not to be encyclopedic, but rather to lay a simple background in which to phrase some of the key motivating results in the area.

Two of the basic formalisms discussed in this paper are timed automata (both ordinary and alternating) and metric logics. Timed automata are most commonly given a semantics in terms of *timed words*, i.e., sequences of instantaneous, real-valued timestamped events, whereas metric logics are more naturally predicated on piecewise-continuous *flows* or *signals*. Accordingly, these are the semantics we adopt here; this does not prevent us from specifying timed-automaton behaviours using metric logics, as timed words can naturally be viewed as particular kinds of flows.

In this paper, we are largely concerned with behaviours over time domains of the form $[0, N)$, where $N \in \mathbb{N}$ is some fixed positive integer. Let us therefore in general write $\mathbb{T}$ to denote either $[0, N)$ or $\mathbb{R}_{\geq 0}$.

Let Σ denote a finite set (or *alphabet*) or *events*. Typical elements of Σ are written a, b, c, a_1, etc. A ***timed word*** is a pair (σ, τ), where $\sigma = \langle a_1 a_2 \ldots a_n \rangle \in \Sigma^*$ is a finite word and $\tau = \langle t_1 t_2 \ldots t_n \rangle \in \mathbb{T}^*$ is a strictly increasing sequence of real-valued *timestamps* of the same length.[1] Note that while we are restricting ourselves to finite timed words, there is no *a priori* bound on the number of events.

Let $\mathbf{MP}$ be a set of *monadic predicates*, denoted P, Q, R, etc. Monadic predicates will alternately be viewed as second-order variables over $\mathbb{T}$, i.e., ranging over sets of non-negative real numbers, and as atomic propositions holding at various points in time. Given $\mathbf{P} \subseteq \mathbf{MP}$ a finite set of monadic predicates, a ***flow*** (or *signal*) over $\mathbf{P}$ is a function $f : \mathbb{T} \to \mathcal{P}(\mathbf{P})$ that is *finitely variable*. Finite variability is the requirement that the restriction of f to any finite subinterval of $\mathbb{T}$ have only finitely many discontinuities.[2]

A flow $f : \mathbb{T} \to \mathcal{P}(\mathbf{P})$ corresponds to an interpretation of the monadic predicates in $\mathbf{P}$: for any $P \in \mathbf{P}$, the interpretation of P as a subset of $\mathbb{T}$ is simply $\{t \in \mathbb{T} \mid P \in f(t)\}$. Conversely, any (finitely-variable) interpretation of all the predicates in $\mathbf{P}$ defines a unique flow $f : \mathbb{T} \to \mathcal{P}(\mathbf{P})$.

Finally, note that a timed word $(\langle a_1 \ldots a_n \rangle, \langle t_1 \ldots t_n \rangle)$ over alphabet Σ can be viewed as a (particular type of) flow, as follows. Let $\mathbf{P} = \Sigma$, and set $f(t_i) = \{a_i\}$, for $1 \leq i \leq n$, and $f(t) = \emptyset$ for all other values of $t \in \mathbb{T}$.

[1] This gives rise to the so-called *strongly monotonic* semantics; in contrast, the *weakly monotonic* semantics allows multiple events to happen 'simultaneously' (or, more precisely, with null-duration delays between them).

[2] It is commonly argued that infinitely-variable flows do not correspond to 'feasible' computations, hence the above restriction. It is however important to stress that we do not place any *a priori* bound on the variability (unlike, for example, [54]), other than requiring that it be finite.

3 Timed Automata

As discussed in the Introduction, we treat Alur-Dill timed automata, interpreted over finite timed words, as the central theoretical implementation formalism in this work.

Let X be a finite set of clocks, denoted x, y, z, etc. We define the set Φ_X of clock constraints over X via the following grammar, where $k \in \mathbb{N}$ stands for any non-negative integer, and $\bowtie \in \{=, \neq, <, >, \leq, \geq\}$ is a comparison operator:

$$\phi ::= \mathbf{true} \;|\; \mathbf{false} \;|\; x \bowtie k \;|\; x - y \bowtie k \;|\; \phi_1 \wedge \phi_2 \;|\; \phi_1 \vee \phi_2 \,.$$

A ***timed automaton*** $\mathcal{A}$ is a six-tuple $(\Sigma, S, S_I, S_F, X, \delta)$, where:

- Σ is a finite set of events,
- S is a finite set of states,
- $S_I \subseteq S$ is a set of initial states,
- $S_F \subseteq S$ is a set of accepting states,
- X is a finite set of clocks, and
- $\delta : S \times \Sigma \times \Phi_X \to \mathcal{P}(S \times \mathcal{P}(X))$ is the transition function: if $(s', R) \in \delta(s, a, \phi)$, then $\mathcal{A}$ allows a jump from state s to state s', consuming event a in the process, provided the constraint ϕ on clocks is met. Afterwards, the clocks in R are reset to zero, while all other clocks remain unchanged. We require that δ be finite, in the sense of having only finitely many inputs not mapping to $\emptyset$.

Given a timed automaton $\mathcal{A}$ as above, a *clock valuation* is a function $\nu : X \to \mathbb{R}_{\geq 0}$. If $t \in \mathbb{R}_{\geq 0}$, we let $\nu + t$ be the clock valuation such that $(\nu + t)(x) = \nu(x) + t$ for all $x \in X$.

A *configuration* of $\mathcal{A}$ is a pair (s, ν), where $s \in S$ is a state and ν is a clock valuation.

An *accepting run* of $\mathcal{A}$ is a finite alternating sequence of configurations and delayed transitions $\pi = (s_0, \nu_0) \xrightarrow{d_1, a_1} (s_1, \nu_1) \xrightarrow{d_2, a_2} \ldots \xrightarrow{d_n, a_n} (s_n, \nu_n)$, with each $d_i \in \mathbb{R}_{>0}$ and $a_i \in \Sigma$, subject to the following conditions:

1. $s_0 \in S_I$, and for all $x \in X$, $\nu_0(x) = 0$,
2. for each $1 \leq i \leq n$, there are some $R_i \subseteq X$ and $\phi_i \in \Phi_X$ such that: (i) $\nu_{i-1} + d_i$ satisfies ϕ_i, (ii) $(s_i, R_i) \in \delta(s_{i-1}, a_i, \phi_i)$, and (iii) $\nu_i(x) = \nu_{i-1}(x) + d_i$ for all $x \in X \setminus R_i$, and $\nu_i(x) = 0$ for all $x \in R_i$, and
3. $s_n \in S_F$.

Each d_i is interpreted as the (strictly positive) time delay between the firing of transitions, and each configuration (s_i, ν_i), for $i \geq 1$, records the data immediately following the i^{th} transition.

A timed word $(\langle a_1 a_2 \ldots a_n \rangle, \langle t_1 t_2 \ldots t_n \rangle)$ is *accepted* by $\mathcal{A}$ if $\mathcal{A}$ has some accepting run of the form $\pi = (s_0, \nu_0) \xrightarrow{d_1, a_1} (s_1, \nu_1) \xrightarrow{d_2, a_2} \ldots \xrightarrow{d_n, a_n} (s_n, \nu_n)$ where, for each $1 \leq i \leq n$, $t_i = d_1 + d_2 + \ldots + d_i$.

Finally, given time domain $\mathbb{T}$, we write $L_{\mathbb{T}}(\mathcal{A})$ to denote the language of $\mathcal{A}$ over $\mathbb{T}$, i.e., the set of timed words accepted by $\mathcal{A}$ all of whose timestamps belong to $\mathbb{T}$.

An example of a timed automaton is provided in Fig. 1, along with a description of its accepted language in the surrounding text.

4 Metric Logics

We introduce *metric* (or *quantitative*) logics to reason about and specify real-time behaviours. We consider both predicate and temporal formalisms, and investigate their relative expressiveness in Sec. 6.

Let **Var** be a set of *first-order variables*, denoted x, y, z, etc., ranging over $\mathbb{T}$. *Second-order monadic formulas* are obtained from the following grammar:

$$\varphi ::= \mathbf{true} \mid x < y \mid +1(x,y) \mid P(x) \mid \varphi_1 \land \varphi_2 \mid \varphi_1 \lor \varphi_2 \mid \neg\varphi \mid \forall x\, \varphi \mid \forall P\, \varphi\,,$$

where $P \in \mathbf{MP}$ is a monadic predicate (viewed here as a second-order variable over $\mathbb{T}$), and $+1$ is a binary relation symbol, with the intuitive interpretation of $+1(x,y)$ as '$x + 1 = y$'.[3] We refer to $\forall x$ and $\forall P$ as *first-order* and *second-order* quantifiers respectively. Existential quantifiers $\exists x$ and $\exists P$ are definable via standard dualities.

The ***monadic second-order metric logic of order***, written $\mathsf{MSO}(<, +1)$, comprises all second-order monadic formulas. Its first-order fragment, the ***(monadic) first-order metric logic of order***, written $\mathsf{FO}(<, +1)$, comprises all $\mathsf{MSO}(<, +1)$ formulas that do not contain any second-order quantifier; note that these formulas are however allowed free monadic predicates.

We also define two further purely order-theoretic sublogics, which are peripheral to our main concerns but necessary to express some key related results. The *monadic second-order logic of order*, $\mathsf{MSO}(<)$, comprises all second-order monadic formulas that do not make use of the $+1$ relation. Likewise, the *(monadic) first-order logic of order*, $\mathsf{FO}(<)$, comprises those $\mathsf{MSO}(<)$ formulas that do not figure second-order quantification.

Metric Temporal Logic, abbreviated MTL, comprises the following *temporal formulas*:

$$\theta ::= \mathbf{true} \mid P \mid \theta_1 \land \theta_2 \mid \theta_1 \lor \theta_2 \mid \neg\theta \mid \Diamond_I \theta \mid \Box_I \theta \mid \theta_1 \,\mathcal{U}_I\, \theta_2\,,$$

where $P \in \mathbf{MP}$ is a monadic predicate (viewed here as an atomic proposition), and $I \subseteq \mathbb{R}_{\geq 0}$ is an open, closed, or half-open interval with endpoints in $\mathbb{N} \cup \{\infty\}$. If $I = [0, \infty)$, then we omit the annotation I in the corresponding temporal operator.

[3] The usual approach is of course to define $+1$ as a unary function symbol; this however necessitates an awkward treatment over bounded domains, as considered in this paper. We shall nonetheless abuse notation later on and invoke $+1$ as if it were a function, in the interest of clarity.

Finally, *Linear Temporal Logic*, written LTL, consists of those MTL formulas in which every indexing interval I on temporal operators is $[0, \infty)$ (and hence omitted).

Figure 2 pictorially summarises the syntactic inclusions and relative expressive powers of these various logics.

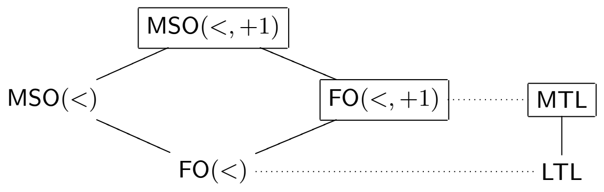

Fig. 2. Relative expressiveness among the various logics. Metric logics are enclosed in boxes. Straight lines denote syntactical inclusion, whereas dotted lines indicate expressive equivalence over bounded time domains (cf. Sec. 6).

We now ascribe a semantics to these various logics in terms of flows over $\mathbb{T}$. Given a formula φ of $\mathsf{MSO}(<, +1)$ or one of its sublogics, let $\mathbf{P}$ and $\{x_1, \ldots, x_n\}$ respectively be the sets of free monadic predicates and free first-order variables appearing in φ. For any flow $f : \mathbb{T} \to \mathcal{P}(\mathbf{P})$ and real numbers $a_1, \ldots, a_n \in \mathbb{T}$, the satisfaction relation $(f, a_1, \ldots, a_n) \models \varphi$ is defined inductively on the structure of φ in the standard way. For example:

- $(f, a) \models P(x)$ iff $P \in f(a)$.
- $(f, a_1, \ldots, a_n) \models \forall P\, \varphi$ iff for all flows $g : \mathbb{T} \to \mathcal{P}(\mathbf{P} \cup \{P\})$ extending f (i.e., such that $g\restriction_{\mathbf{P}} = f$), we have $(g, a_1, \ldots, a_n) \models \varphi$.
  (Here $\mathbf{P}$ is the set of free monadic predicates appearing in $\forall P\, \varphi$, and therefore does not contain P.)

And so on.

We shall particularly be interested in the special case in which φ is a *sentence*, i.e., a formula with no free first-order variable. In such instances, we simply write the satisfaction relation as $f \models \varphi$.

For θ an MTL or LTL formula, let $\mathbf{P}$ be the set of monadic predicates appearing in θ. Given a flow $f : \mathbb{T} \to \mathcal{P}(\mathbf{P})$ and $t \in \mathbb{T}$, the satisfaction relation $(f, t) \models \theta$ is defined inductively on the structure of θ, as follows:

- $(f, t) \models \mathbf{true}$.
- $(f, t) \models P$ iff $P \in f(t)$.
- $(f, t) \models \theta_1 \wedge \theta_2$ iff $(f, t) \models \theta_1$ and $(f, t) \models \theta_2$.
- $(f, t) \models \theta_1 \vee \theta_2$ iff $(f, t) \models \theta_1$ or $(f, t) \models \theta_2$.
- $(f, t) \models \neg\theta$ iff $(f, t) \not\models \theta$.
- $(f, t) \models \Diamond_I \theta$ iff there exists $u \in \mathbb{T}$ with $u > t$, $u - t \in I$, and $(f, u) \models \theta$.
- $(f, t) \models \Box_I \theta$ iff for all $u \in \mathbb{T}$ with $u > t$ and $u - t \in I$, $(f, u) \models \theta$.

- $(f, t) \models \theta_1 \, \mathcal{U}_I \, \theta_2$ iff there exists $u \in \mathbb{T}$ with $u > t$, $u - t \in I$, $(f, u) \models \theta_2$, and for all $v \in (t, u)$, $(f, v) \models \theta_1$.

Finally, we write $f \models \theta$ iff $(f, 0) \models \theta$. This is sometimes referred to as the *initial semantics*.

Note that we have adopted a *strict* semantics, in which the present time t has no influence on the truth values of future temporal subformulas.

An important point concerning our semantics is that it is *continuous*, rather than *pointwise*: more precisely, the temporal operators quantify over all time points of the domain, as opposed to merely those time points at which discontinuities occur. Positive decidability results for satisfiability and model checking of MTL over unbounded time intervals have been obtained in the pointwise semantics [44, 45, 46]; it is worth noting that none of these results hold in the continuous semantics.

5 Alternating Timed Automata

We introduce alternating timed automata as a generalisation of ordinary timed automata in which, in addition to *disjunctive* (or nondeterministic) transitions, one also allows *conjunctive* transitions. Our notation closely follows that of Sec. 3.

For *Prop* a set of propositional variables, the collection $\mathcal{B}_+(Prop)$ of positive Boolean formulas over *Prop* is given by the following grammar:

$$\psi ::= \mathbf{true} \mid \mathbf{false} \mid p \mid \psi_1 \vee \psi_2 \mid \psi_1 \wedge \psi_2 \,,$$

where $p \in Prop$. A subset $M \subseteq Prop$ *satisfies* $\psi \in \mathcal{B}_+(Prop)$ if the truth assignment that ascribes **true** to elements of M and **false** to elements of $Prop \setminus M$ satisfies ψ.

An ***alternating timed automaton*** is a six-tuple $\mathcal{A} = (\Sigma, S, S_I, S_F, X, \delta)$, where

- Σ is a finite set of events,
- S is a finite set of states,
- $S_I \subseteq S$ is a set of initial states,
- $S_F \subseteq S$ is a set of accepting states,
- X is a finite set of clocks, and
- $\delta : S \times \Sigma \times \Phi_X \to \mathcal{B}_+(S \times \mathcal{P}(X))$ is the transition function. We require that δ be finite, in the sense of having only finitely many inputs not mapping to **false**.

Intuitively, the transition function of an alternating timed automaton is interpreted as follows: in state (s, ν), if ν satisfies the clock constraint ϕ and $\{(s_1, R_1), \ldots, (s_k, R_k)\}$ satisfies $\delta(s, a, \phi)$, then we think of the automaton as having a *conjunctive transition* $(s, \nu) \xrightarrow{a} \{(s_1, \nu_1), \ldots, (s_k, \nu_k)\}$, where each clock valuation ν_i is the same as ν except for the clocks in R_i which are all reset to zero.

As an example, let us define an automaton $\mathcal{A}$ over alphabet $\Sigma = \{a\}$ that accepts those words such that for every timed event (a, t) with $t < 1$ there is an event $(a, t + 1)$ exactly one time unit later. $\mathcal{A}$ has a single clock x and set of locations $\{s, u\}$, with s initial and accepting, and u non-accepting. The transition function is defined by:

$$\begin{aligned} &\delta(s, a, x < 1) = (s, \emptyset) \wedge (u, \{x\}) \qquad && \delta(s, a, x \geq 1) = (s, \emptyset) \\ &\delta(u, a, x \neq 1) = (u, \emptyset) && \delta(u, a, x = 1) = \mathbf{true} \end{aligned}$$

The automaton is illustrated in Fig. 3 in which we represent the conjunctive transition by connecting two arrows with an arc.

A run of $\mathcal{A}$ starts in location s. Every time an a occurs in the first time unit, the automaton makes a simultaneous transition to both s and u, thus opening up a new thread of computation equipped with a fresh copy of the clock x. The automaton must eventually leave location u, which is non-accepting, and it can only do so exactly one time unit after first entering the location.

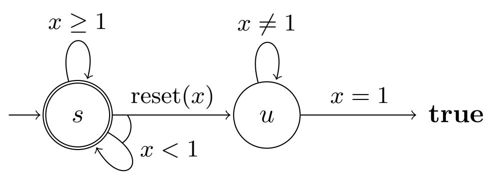

Fig. 3. Alternating timed automaton $\mathcal{A}$.

We now formally define the language accepted by an alternating timed automaton $\mathcal{A} = (\Sigma, S, S_I, S_F, X, \delta)$. A run of $\mathcal{A}$ over a timed word $(\langle a_1 a_2 \dots a_n \rangle, \langle t_1 t_2 \dots t_n \rangle)$ is a finite dag satisfying the following conditions: (i) each vertex is a triple (i, s, ν), with $0 \leq i \leq n$, $s \in S$ a location, and ν a clock valuation; (ii) there is a vertex $(0, s_0, \nu_0)$, where $s_0 \in S_I$ and $\nu_0(x) = 0$ for all $x \in X$; (iii) each vertex (i, s, ν), $i \leq n - 1$, has a (possibly empty) set of children of the form $\{(i + 1, s_1, \nu_1), \dots, (i + 1, s_k, \nu_k)\}$ where, writing $\nu' = \nu + t_{i+1} - t_i$ (and adopting the convention that $t_0 = 0$), there is a conjunctive transition $(s, \nu') \xrightarrow{a_i} \{(s_1, \nu_1), \dots, (s_k, \nu_k)\}$.

The run is *accepting* if for each vertex (n, s, ν), s is an accepting location; in this case we say that the timed word $(\langle a_1 a_2 \dots a_n \rangle, \langle t_1 t_2 \dots t_n \rangle)$ is *accepted* by $\mathcal{A}$. The language $L_{\mathbb{T}}(\mathcal{A})$ of $\mathcal{A}$ over $\mathbb{T}$ is the set of timed words accepted by $\mathcal{A}$ all of whose timestamps belong to $\mathbb{T}$.

One of the motivations for introducing alternating timed automata is that they enjoy better closure properties than ordinary timed automata:

Proposition 1. *For any time domain $\mathbb{T}$, alternating timed automata are effectively closed under union, intersection, and complement [35, 46].*

6 Expressiveness

Fix a time domain $\mathbb{T}$, and let $\mathcal{L}$ and $\mathcal{J}$ be two logics. We say that $\mathcal{L}$ is ***at least as expressive as*** $\mathcal{J}$ if, for any sentence θ of $\mathcal{J}$, there exists a sentence φ of $\mathcal{L}$ such that θ and φ are satisfied by precisely the same set of flows over $\mathbb{T}$.

Two logics are then said to be ***equally expressive*** if each is at least as expressive as the other.

The following result can be viewed as an extension of Kamp's celebrated theorem, asserting the expressive equivalence of $\mathsf{FO}(<)$ and LTL [29, 19], to metric logics over bounded time domains:

Theorem 1. *For any fixed bounded time domain of the form* $[0, N)$*, with* $N \in \mathbb{N}$*, the metric logics* $\mathsf{FO}(<, +1)$ *and* MTL *are equally expressive. Moreover, this equivalence is effective [40].*

Note that expressiveness here is relative to a *single* structure $\mathbb{T}$, rather than to a *class* of structures. In particular, although $\mathsf{FO}(<, +1)$ and MTL are equally expressive over any bounded time domain of the form $[0, N)$, the correspondence and witnessing formulas may very well vary according to the time domain.

It is interesting to note that $\mathsf{FO}(<, +1)$ is strictly more expressive than MTL over $\mathbb{R}_{\geq 0}$ [27, 11]. For example, MTL is incapable of expressing the following formula (in slightly abusive but readable notation)

$$\exists x \, \exists y \, \exists z \, (x < y < z < x + 1 \wedge P(x) \wedge P(y) \wedge P(z))$$

over the non-negative reals. This formula asserts that, sometime in the future, P will hold at three distinct time points within a single time unit.

It is also worth noting that $\mathsf{MSO}(<, +1)$ is strictly more expressive than $\mathsf{FO}(<, +1)$—and hence MTL—over any time domain.

7 Decision Problems

We now turn to various decision problems concerning timed automata and metric logics over bounded time domains. Recall that the latter are real intervals of the form $[0, N)$, with $N \in \mathbb{N}$ considered part of the input (written in binary). We also contrast our results with their known counterparts over the non-negative reals $\mathbb{R}_{\geq 0}$.

The most fundamental verification question is undoubtedly the ***emptiness*** (or *reachability*) ***problem***: does a given timed automaton accept some timed word? It is well known that the problem is PSPACE-complete over $\mathbb{R}_{\geq 0}$ [1], and the proof is easily seen to carry over to bounded time domains:

Theorem 2. *The time-bounded emptiness problem for timed automata is PSPACE-complete (following [1]).*

The ***language-inclusion problem*** takes as inputs two timed automata, $\mathcal{A}$ and $\mathcal{B}$, sharing a common alphabet, and asks whether every timed word accepted by $\mathcal{A}$ is also accepted by $\mathcal{B}$. Unfortunately, language inclusion is undecidable over $\mathbb{R}_{\geq 0}$ [2]. However:

Theorem 3. *The time-bounded language-inclusion problem for timed automata is decidable and 2EXPSPACE-complete [40].*

For a fixed metric logic $\mathcal{L}$, the ***satisfiability problem*** asks, given a sentence φ of $\mathcal{L}$ over a set $\mathbf{P}$ of free monadic predicates, whether there exists a flow over $\mathbf{P}$ satisfying φ. The ***model-checking problem*** for $\mathcal{L}$ takes as inputs a timed automaton $\mathcal{A}$ over alphabet Σ, together with a sentence φ of $\mathcal{L}$ with set of free monadic predicates $\mathbf{P} = \Sigma$, and asks whether every timed word (viewed as a flow) accepted by $\mathcal{A}$ satisfies φ.

The canonical time domain for interpreting the metric logics $\mathsf{MSO}(<, +1)$, $\mathsf{FO}(<, +1)$, and MTL is the non-negative real line $\mathbb{R}_{\geq 0}$. Unfortunately, none of these logics are decidable over $\mathbb{R}_{\geq 0}$ [5, 6, 26]. The situation however differs over bounded time domains, as the following result indicates:

Theorem 4. *The time-bounded satisfiability and model-checking problems for the metric logics* $\mathsf{MSO}(<, +1)$*,* $\mathsf{FO}(<, +1)$*, and* MTL *are all decidable, with the following complexities [40]:*

$\mathsf{MSO}(<, +1)$	Non-elementary
$\mathsf{FO}(<, +1)$	Non-elementary
MTL	EXPSPACE-complete

Finally, we turn our attention to alternating timed automata. The ***emptiness*** and ***language-inclusion problems*** are of course defined in the same way as for ordinary timed automata. One may also wish to model check a timed automaton $\mathcal{A}$ (*qua* implementation) against an alternating timed automaton $\mathcal{B}$ (*qua* specification). Note that since alternating timed automata are closed under all Boolean operations (in linear time), these problems are all polynomial-time equivalent to the emptiness problem.

Unfortunately, emptiness is undecidable for alternating timed automata over $\mathbb{R}_{\geq 0}$, by immediate reduction from the undecidability of language inclusion for ordinary timed automata. Thankfully, the situation over bounded time domains is more favourable:

Theorem 5. *The time-bounded emptiness and language-inclusion problems for alternating timed automata are decidable, but with non-elementary complexity [28].*

8 Discussion and Future Directions

In this work, we have attempted to promote a theory of time-bounded verification to answer, at least in part, Trakhtenbrot's 15-year-old challenge to 'lift the classical theory to the real-time world'. We have argued that this theory is both ***pertinent***, in that it is fully adequate to handle a large proportion of 'real-world' real-time systems and specifications; and ***effective***, in that the restriction to bounded time domains reclaims as decidable several of the key decision problems of real-time verification.

In terms of future work, we list below a sample of possible research avenues, roughly divided along four main axes:

I. Extensions to further real-time formalisms. In this paper, we have entirely focussed on *linear-time* semantics. Of course, a great deal of classical and real-time verification work has been carried out in *branching-time* settings, and it would be interesting to investigate whether the time-bounded approach can be usefully combined with branching-time paradigms. Several researchers have also considered various extensions of timed automata, such as *weighted timed automata* and *hybrid automata*, and assorted verification problems; again, reformulating relevant questions in a time-bounded context may prove fruitful. Another direction is that of *timed games* and related topics such as *timed controller synthesis*.

II. Algorithmic and complexity issues. The complexity bounds presented in this paper are fairly coarse-grained. In many instances, a finer 'parameterised' analysis (in which one or more of the inputs, such as the time domain, are considered fixed) would undoubtedly yield valuable additional insight. Another promising direction is to investigate combining existing algorithmic techniques, such as those exploiting *flatness* in MTL formulas [13], with the algorithms specific to time-bounded verification.

III. Expressiveness. Many important questions regarding expressiveness are left entirely unanswered. Do metric logics have equivalent alternating timed automata counterparts, and vice-versa? Can one develop an attractive theory of *timed regular expressions* over bounded time domains? Is there a good notion of *robustness* for time-bounded languages, in the sense of being impervious to sufficiently small perturbations in the timestamps?

IV. Implementation and case studies. The history of verification has been marked by a mutually beneficial interaction of theory and practice. We believe it would be highly desirable, in conjunction with the study of the theoretical concerns discussed here, to evaluate the *practical* effectiveness of time-bounded verification on real-world examples. This will no doubt require the development of appropriate abstraction schemes, data structures, symbolic techniques, algorithmic heuristics, etc. Ultimately, however, a time-bounded theory of verification can only gain widespread acceptance if its usefulness is adequately demonstrated.

References

[1] R. Alur, C. Courcoubetis, and D. Dill. Model-checking for real-time systems. In *Proceedings of LICS*. IEEE Computer Society Press, 1990.

[2] R. Alur and D. Dill. A theory of timed automata. *Theor. Comput. Sci.*, 126, 1994.

[3] R. Alur, T. Feder, and T. A. Henzinger. The benefits of relaxing punctuality. *J. ACM*, 43(1), 1996.

[4] R. Alur, L. Fix, and T. A. Henzinger. Event-clock automata: A determinizable class of timed automata. *Theor. Comput. Sci.*, 211, 1999.

[5] R. Alur and T. A. Henzinger. Logics and models of real time: A survey. In *Proceedings of REX Workshop*, volume 600 of *LNCS*. Springer, 1991.

[6] R. Alur and T. A. Henzinger. Real-time logics: Complexity and expressiveness. *Inf. Comput.*, 104(1), 1993.

[7] R. Alur and T. A. Henzinger. A really temporal logic. *J. ACM*, 41(1), 1994.

[8] R. Alur, S. La Torre, and P. Madhusudan. Perturbed timed automata. In *Proceedings of HSCC*, volume 3414 of *LNCS*. Springer, 2005.

[9] E. Asarin, P. Caspi, and O. Maler. Timed regular expressions. *J. ACM*, 49(2), 2002.

[10] C. Baier, H. Hermanns, J.-P. Katoen, and B. R. Haverkort. Efficient computation of time-bounded reachability probabilities in uniform continuous-time Markov decision processes. *Theor. Comput. Sci.*, 345(1), 2005.

[11] P. Bouyer, F. Chevalier, and N. Markey. On the expressiveness of TPTL and MTL. In *Proceedings of FSTTCS*, volume 3821 of *LNCS*. Springer, 2005.

[12] P. Bouyer, N. Markey, J. Ouaknine, and J. Worrell. The cost of punctuality. In *Proceedings of LICS*. IEEE Computer Society Press, 2007.

[13] P. Bouyer, N. Markey, J. Ouaknine, and J. Worrell. On expressiveness and complexity in real-time model checking. In *Proceedings of ICALP*, volume 5126 of *LNCS*. Springer, 2008.

[14] D. Bošnački. Digitization of timed automata. In *Proceedings of FMICS*, 1999.

[15] A. K. Chandra, D. Kozen, and L. J. Stockmeyer. Alternation. *J. ACM*, 28(1), 1981.

[16] M. Dickhöfer and T. Wilke. Timed alternating tree automata: The automata-theoretic solution to the TCTL model checking problem. In *Proceedings of ICALP*, volume 1644 of *LNCS*. Springer, 1999.

[17] E. A. Emerson and C. S. Jutla. Tree automata, mu-calculus and determinacy (extended abstract). In *Proceedings of FOCS*. IEEE Computer Society Press, 1991.

[18] M. Emmi and R. Majumdar. Decision problems for the verification of real-time software. In *Proceedings of HSCC*, volume 3927 of *LNCS*. Springer, 2006.

[19] D. M. Gabbay, A. Pnueli, S. Shelah, and J. Stavi. On the temporal basis of fairness. In *Proceedings of POPL*. ACM Press, 1980.

[20] V. Gupta, T. A. Henzinger, and R. Jagadeesan. Robust timed automata. In *Proceedings of HART*, volume 1201 of *LNCS*. Springer, 1997.

[21] T. A. Henzinger. *The Temporal Specification and Verification of Real-Time Systems*. PhD thesis, Stanford University, 1991. Technical Report STAN-CS-91-1380.

[22] T. A. Henzinger, Z. Manna, and A. Pnueli. What good are digital clocks? In *Proceedings of ICALP*, volume 623 of *LNCS*. Springer, 1992.

[23] T. A. Henzinger and J.-F. Raskin. Robust undecidability of timed and hybrid systems. In *Proceedings of HSCC*, volume 1790 of *LNCS*. Springer, 2000.

[24] T. A. Henzinger, J.-F. Raskin, and P.-Y. Schobbens. The regular real-time languages. In *Proceedings of ICALP*, volume 1443 of *LNCS*. Springer, 1998.

[25] P. Herrmann. Timed automata and recognizability. *Inf. Process. Lett.*, 65, 1998.

[26] Y. Hirshfeld and A. Rabinovich. Logics for real time: Decidability and complexity. *Fundam. Inform.*, 62(1), 2004.

[27] Y. Hirshfeld and A. Rabinovich. Expressiveness of metric modalities for continuous time. *Logical Methods in Computer Science*, 3(1), 2007.

[28] M. Jenkins, J. Ouaknine, A. Rabinovich, and J. Worrell. Alternating timed automata over bounded time. In *Proceedings of LICS*. IEEE Computer Society Press, 2010.

[29] H. Kamp. *Tense Logic and the Theory of Linear Order*. PhD thesis, University of California, 1968.

[30] J.-P. Katoen and I. S. Zapreev. Safe on-the-fly steady-state detection for time-bounded reachability. In *Proceedings of QEST*. IEEE Computer Society Press, 2006.

[31] D. K. Kaynar, N. Lynch, R. Segala, and F. Vaandrager. Timed I/O Automata: A mathematical framework for modeling and analyzing real-time systems. In *Proceedings of RTSS*. IEEE Computer Society Press, 2003.

[32] R. Koymans. Specifying real-time properties with metric temporal logic. *Real-Time Systems*, 2(4), 1990.

[33] O. Kupferman and M. Y. Vardi. Weak alternating automata are not that weak. *ACM Trans. Comput. Log.*, 2(3), 2001.

[34] S. Lasota and I. Walukiewicz. Alternating timed automata. In *Proceeding of FOSSACS*, volume 3441 of *LNCS*. Springer, 2005.

[35] S. Lasota and I. Walukiewicz. Alternating timed automata. *ACM Trans. Comput. Log.*, 9(2), 2008.

[36] C. Löding and W. Thomas. Alternating automata and logics over infinite words. In *Proceedings of IFIP TCS*, volume 1872 of *LNCS*. Springer, 2000.

[37] N. A. Lynch and H. Attiya. Using mappings to prove timing properties. *Distributed Computing*, 6(2), 1992.

[38] J. Ostroff. *Temporal Logic of Real-Time Systems*. Research Studies Press, 1990.

[39] J. Ouaknine. Digitisation and full abstraction for dense-time model checking. In *Proceedings of TACAS*, volume 2280 of *LNCS*. Springer, 2002.

[40] J. Ouaknine, A. Rabinovich, and J. Worrell. Time-bounded verification. In *Proceedings of CONCUR*, volume 5710 of *LNCS*. Springer, 2009.

[41] J. Ouaknine and J. Worrell. Revisiting digitization, robustness, and decidability for timed automata. In *Proceedings of LICS*. IEEE Computer Society Press, 2003.

[42] J. Ouaknine and J. Worrell. Universality and language inclusion for open and closed timed automata. In *Proceedings of HSCC*, volume 2623 of *LNCS*. Springer, 2003.

[43] J. Ouaknine and J. Worrell. On the language inclusion problem for timed automata: Closing a decidability gap. In *Proceedings of LICS*. IEEE Computer Society Press, 2004.

[44] J. Ouaknine and J. Worrell. On the decidability of Metric Temporal Logic. In *Proceedings of LICS*. IEEE Computer Society Press, 2005.

[45] J. Ouaknine and J. Worrell. Safety Metric Temporal Logic is fully decidable. In *Proceedings of TACAS*, volume 3920 of *LNCS*. Springer, 2006.

[46] J. Ouaknine and J. Worrell. On the decidability and complexity of Metric Temporal Logic over finite words. *Logical Methods in Computer Science*, 3(1), 2007.

[47] J.-F. Raskin. *Logics, Automata and Classical Theories for Deciding Real Time*. PhD thesis, University of Namur, 1999.

[48] J.-F. Raskin and P.-Y. Schobbens. State-clock logic: A decidable real-time logic. In *Proceedings of HART*, volume 1201 of *LNCS*. Springer, 1997.

[49] O. Roux and V. Rusu. Verifying time-bounded properties for ELECTRE reactive programs with stopwatch automata. In *Proceedings of Hybrid Systems*, volume 999 of *LNCS*. Springer, 1994.

[50] S. Taşiran, R. Alur, R. P. Kurshan, and R. K. Brayton. Verifying abstractions of timed systems. In *Proceedings of CONCUR*, volume 1119 of *LNCS*. Springer, 1996.

[51] B. A. Trakhtenbrot. Origins and metamorphoses of the trinity: Logic, nets, automata. In *Proceedings of LICS*. IEEE Computer Society Press, 1995.

[52] M. Y. Vardi. Alternating automata and program verification. In *Computer Science Today*, volume 1000 of *LNCS*. Springer, 1995.

[53] M. Y. Vardi. From philosophical to industrial logics. In *Proceedings of ICLA*, volume 5378 of *LNCS*. Springer, 2009.

[54] T. Wilke. Specifying timed state sequences in powerful decidable logics and timed automata. In *Proceedings of FTRTFT*, volume 863 of *LNCS*. Springer, 1994.