International Journal of Foundations of Computer Science
© World Scientific Publishing Company

A LOGICAL CHARACTERISATION OF EVENT CLOCK AUTOMATA*

DEEPAK D'SOUZA
Chennai Mathematical Institute,
92 G. N. Chetty Road, Chennai 600 017 India.

Received (received date)
Revised (revised date)
Communicated by Editor's name

ABSTRACT

We show that the class of Event Clock Automata [2] admit a logical characterisation via an *unrestricted* monadic second order logic interpreted over timed words. The result is interesting in that it provides an unrestricted yet decidable logical characterisation of a non-trivial class of timed languages. A timed temporal logic considered earlier in the literature [11] is shown to be expressively complete with respect to the monadic logic.

Keywords: verification of real-time systems, timed automata, logic in computer science, temporal logic

1. Introduction

The timed automata of Alur and Dill [1] are a standard model for describing timed behaviours. They augment classical automata with clocks which can be read and reset while taking transitions. These automata are very powerful in language theoretic terms: while their languages are closed under union and intersection and their emptiness problem is decidable, their languages are not closed under complementation and their language inclusion problem is undecidable. Consequently, the verification problem, which is often phrased as whether $L(\mathcal{A}_{Prog}) \subseteq L(\mathcal{A}_{Spec})$, cannot be solved for these automata in general. One must then either work with deterministic specifications or with a restricted class of timed automata which has the required closure properties.

The *event clock automata* of Alur, Fix, and Henzinger [2] are a subclass of timed automata which permit non-determinism (which is useful from a modelling point of view), and are closed under boolean operations, including complementation. The key feature of these automata is that they have a pair of *implicit* clocks associated with each action. The clocks record the time elapsed since the last occurrence of the associated action, as well as the time that *will* elapse before the next occurrence. As a result common real-time requirements like bounded response time can be naturally

*A preliminary version of this paper appeared as [5] modelled using these automata.

In this paper we argue in favour of these automata from a logical viewpoint. In the classical setting, the existence of a monadic second order logical characterisation for a class of languages is a strong endorsement of its regularity. Such a characterisation can also help in identifying natural temporal logics which can be expected to have advantages in terms of relatively efficient algorithms for solving their verification problem. As is well-known, Linear Time Temporal Logic (LTL) [10] is expressively equivalent to the first-order fragment of S1S, Büchi's monadic second-order logic which characterises untimed regular languages. LTL is natural to use, and has an exponential time algorithm for deciding its satisfiability problem as against the non-elementary decision procedure for the first-order fragment of S1S. The aim of this paper is to show that event clock automata admit a similar logical framework.

We show that event clock automata can be characterised via a monadic second order logic interpreted over timed words. This logic, called MSO_{ec}, has timed modalities of the form $\lhd_a(x) \in I$ which asserts that with respect to the position x in the timed word under consideration, the time elapsed since the last a action lies in the interval I; and $\rhd_a(x) \in I$ which asserts that the time to the next occurrence of a lies in I. The logic is unrestricted and in particular it allows full existential quantification over set variables (or monadic predicates).

We further show that a timed temporal logic proposed earlier in the literature [11] is expressively complete with respect to our monadic logic, in that it corresponds to the first-order fragment of our logic. The logic, called here LTL_{ec}, has timed modalities similar to the ones above, denoted $\lhd_a \in I$ and $\rhd_a \in I$, which assert that at the current position in a timed word the time elapsed since the last a action (respectively the time to the next a action) lies in the interval I. The satisfiability and model-checking problems for the logic have been solved in [11]. The issue of its expressive completeness was however not addressed, and is pinned down here.

There have been several logical characterisations of timed automata and its subclasses proposed in the literature [14, 9, 11, 6]. Unfortunately these logics are all restricted in their syntax, typically in their use of existential quantification. Removing these restrictions invariably leads to undecidability of the logics. One of the first such characterisations was given by Wilke in [14], where he characterises the class of timed automata via the logic $\mathcal{L}\overleftrightarrow{d}$, the monadic second order logic of relative distance. $\mathcal{L}\overleftrightarrow{d}$ has a restricted syntax due to the fact that timed automata are not closed under complementation. In particular, set variables used in a distance predicate must be existentially quantified only at the beginning of the formula.

In [9] Raskin, Schobbens, and Henzinger propose the class of *recursive event clock automata* and a corresponding monadic logic called MinMaxML. Once again the second-order quantification is restricted as one cannot quantify over set variables which are in the scope of a distance operator. The authors also propose a timed temporal logic called EventCLockTL that is expressively complete with respect to MinMaxML. These results are shown in the setting of the so-called "continuous" time semantics [11], where one is allowed to refer to time points *in between* event occurrences. Here however we use the standard "pointwise" interpretation used in [14].

In [6] a subclass of event clock automata called *product interval automata* is studied. These automata admit a logical characterisation which comprises boolean combinations of "local" monadic second order logic assertions. A corresponding timed temporal logic called TLTL$^{\otimes}$ is identified which is expressively complete with respect to this characterisation. The study of these automata and its extensions lead us in a natural way to the class of event clock automata. The techniques used in the present paper essentially build on the ones used in [6].

An interesting aspect of the characterisation of event clock automata is the apparent discrepancy between unrestricted existential quantification in the logic and non-closure of the class of event clock automata under projection. Existential quantification in a monadic logic usually corresponds to the associated class of languages being closed under the operation of projection, or renaming. Event clock automata are however *not* closed under renaming. In Sec. 4 we explain this phenomenon by showing that existential quantification in our logic essentially corresponds to closure under renaming of a weaker class of event clock automata which we call *quasi* event clock automata.

2. Event Clock Automata

Let $\mathbb{N}$ denote the set of natural numbers $\{0, 1, \ldots\}$, and let $\mathbb{R}^{>0}$ and $\mathbb{Q}^{\geq 0}$ denote the set of positive reals and non-negative rationals respectively. As usual the set of finite and infinite words over an alphabet A will be denoted by A^* and A^ω respectively.

In this paper we deal with infinite timed behaviours. Our results can be easily extended to cover finite timed behaviours as well.

A Büchi automaton over an alphabet A is a structure $\mathcal{A} = (Q, \longrightarrow, Q_{in}, F)$ where Q is a finite set of states, $\longrightarrow \subseteq Q \times A \times Q$ is the transition relation, $Q_{in} \subseteq Q$ is a set of initial states, and $F \subseteq Q$ is a set of accepting states. Let $\alpha \in A^\omega$. A run of $\mathcal{A}$ over α is a map $\rho : \mathbb{N} \to Q$ which satisfies: $\rho(0) \in Q_{in}$ and $\rho(i) \xrightarrow{\alpha(i)} \rho(i+1)$ for every $i \in \mathbb{N}$. We say ρ is an *accepting* run of $\mathcal{A}$ on α if $\rho(i) \in F$ for infinitely many $i \in \mathbb{N}$. The set of words accepted by $\mathcal{A}$, denoted here (for reasons which will soon be clear) as $L_{sym}(\mathcal{A})$, is defined to be the set of words in A^ω on which $\mathcal{A}$ has an accepting run. We term a subset L of A^ω *ω-regular* if $L = L_{sym}(\mathcal{A})$ for some Büchi automaton $\mathcal{A}$ over A.

An *(infinite) timed word* over an alphabet Σ is a member σ of $(\Sigma \times \mathbb{R}^{>0})^\omega$ which satisfies the following conditions. Let $\sigma = (a_0, t_0)(a_1, t_1) \cdots$. Then:

1. (*strict monotonicity*) for each $i \in \mathbb{N}$, $t_i < t_{i+1}$,
2. (*progressiveness*) for each $t \in \mathbb{R}^{>0}$ there exists $i \in \mathbb{N}$ such that $t_i > t$.

Let $T\Sigma^\omega$ denote the set of infinite timed words over Σ. For an infinite timed word σ we will also use the representation of σ as (α, η) where $\alpha \in \Sigma^\omega$ and $\eta : \mathbb{N} \to \mathbb{R}^{>0}$.

We will use rational bounded intervals to specify timing constraints. These intervals are the usual open or closed intervals, except that we require them to be

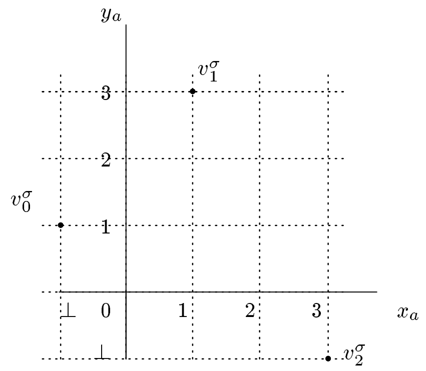

Figure 1: values of x_a and y_a for $\sigma = (a,2)(a,3)(a,6)(b,7)(b,8)\cdots$.

non-empty and the bounds to be rationals. More precisely, these intervals are of the form (l,r), $[l,r)$, $(l,r]$, or $[l,r]$, where $l,r \in \mathbb{Q}^{\geq 0} \cup \{\infty\}$ with $l \leq r$. For an interval of the form $(l,r]$ or $[l,r]$ we require $r \neq \infty$, and for intervals of the form $[l,r)$ or $[l,r]$ we require $l \neq 0$. To avoid empty intervals, unless an interval is of the form $[l,r]$, we require $l < r$. An interval will denote a non-empty, convex subset of reals in the obvious way. For example the interval $[1,\infty)$ denotes the set $\{t \in \mathbb{R}^{>0} \mid 1 \leq t\}$. We also permit the special interval $[\perp,\perp]$ which comprises the single "undefined" value $\perp$. The set of all intervals will be denoted by $\mathcal{I}$.

We now define event clock automata and the timed languages they accept. Our definition is slightly different from the one introduced in [2]. However the definitions can easily be seen to be equivalent. An event clock automaton over an alphabet Σ uses two implicit clocks x_a and y_a for each a in Σ. Along a timed word the clock x_a measures the time since the last occurrence of a, and y_a measures the time *to* the next occurrence of a. The transitions of the automaton are annotated by conditions on these clocks which serve to constrain the time of occurrence of the transition.

Let us fix an alphabet Σ for the rest of this paper. Let C_Σ denote the set of clocks $\{x_a, y_a \mid a \in \Sigma\}$. An *(event clock) interval alphabet* based on Σ is a finite non-empty subset of $\Sigma \times \mathcal{I}^{C_\Sigma}$. Elements of an interval alphabet are thus of the form (a,g) with $a \in \Sigma$ and $g : C_\Sigma \to \mathcal{I}$. The component g is called a *guard* and is meant to represent the constraint: $\bigwedge_{z \in C_\Sigma} z \in g(z)$.

Let $\sigma \in T\Sigma^\omega$ with $\sigma(i) = (a_i, t_i)$ for each $i \in \mathbb{N}$. We define a clock valuation $v_i^\sigma : C_\Sigma \to \mathbb{R}^{>0} \cup \{\perp\}$ which specifies the values of the clocks in C_Σ at position i in σ:

$$v_i^\sigma(x_a) = \begin{cases} t_i - t_j & \text{if } \exists j < i :\ a_j = a \text{ and } \forall k :\ j < k < i \Rightarrow\ a_k \neq a \\ \perp & \text{otherwise.} \end{cases}$$

$$v_i^\sigma(y_a) = \begin{cases} t_j - t_i & \text{if } \exists j > i : a_j = a \text{ and } \forall k :\ i < k < j \Rightarrow a_k \neq a \\ \perp & \text{otherwise.} \end{cases}$$

Figure 1 shows the values of v_i^σ for the clocks x_a and y_a for the timed word $\sigma = (a,2)(a,3)(a,6)(b,7)(b,8)\cdots$.

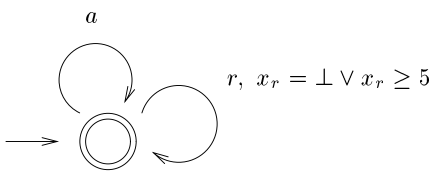

Figure 2: An event clock automaton over $\{a, r\}$.

Let Γ be an interval alphabet based on Σ. Let $\gamma \in \Gamma^\omega$ with $\gamma(i) = (a_i, g_i)$ for each $i \in \mathbb{N}$. Then γ induces in a natural way a set of timed words, denoted $tw(\gamma)$, as follows. Let $\sigma \in T\Sigma^\omega$ with $\sigma(i) = (b_i, t_i)$ for each $i \in \mathbb{N}$. Then $\sigma \in tw(\gamma)$ iff for each $i \in \mathbb{N}$, $b_i = a_i$ and for each $z \in C_\Sigma$, $v_i^\sigma(z) \in g_i(z)$. We extend the map tw to work on subsets of Γ^ω in the natural way. Thus, for $\widehat{L} \subseteq \Gamma^\omega$, we define $tw(\widehat{L}) = \bigcup_{\gamma \in \widehat{L}} tw(\gamma)$.

We are now in a position to define an event clock automaton and the timed language accepted by it. An *event clock automaton* over an alphabet Σ is simply a Büchi automaton over an interval alphabet based on Σ. Viewed as a Büchi automaton over an interval alphabet Γ, an event clock automaton $\mathcal{A}$ accepts the language $L_{sym}(\mathcal{A}) \subseteq \Gamma^\omega$ which we will call the "symbolic" language accepted by $\mathcal{A}$. However, we will be more interested in the timed language accepted by $\mathcal{A}$: this is denoted $L(\mathcal{A})$ and is defined to be $tw(L_{sym}(\mathcal{A}))$. We say that $L \subseteq T\Sigma^\omega$ is a *ω-regular event clock language* over Σ if $L = L(\mathcal{A})$ for some event clock automaton $\mathcal{A}$ over Σ. Thus, ω-regular event clock languages over Σ are precisely those of the form $tw(\widehat{L})$, where $\widehat{L}$ is an ω-regular language over an interval alphabet based on Σ.

Figure 2 shows an event clock automaton over the alphabet $\Sigma' = \{a, r\}$. The timed language accepted by the automaton is the set of time words over Σ' such that consecutive r's (for "requests") are separated by at least 5 units of time. The interval alphabet we have in mind is

$$\Gamma = \{(a, g) \mid g \in G\} \cup \{(r, g') \mid g' \in G'\},$$

where the set of guards

$$G = \{g \mid g(z) = [\perp, \perp] \text{ or } (0, \infty) \text{ for } z = x_a, y_a, x_r, y_r\},$$

and

$$G' = \{g' \mid g'(x_r) = [\perp, \perp] \text{ or } [5, \infty) \text{ and } g'(z) = [\perp, \perp] \text{ or } (0, \infty) \text{ for } z = x_a, y_a, y_r\}.$$

The labels from the interval alphabet have been written using the standard notation for timed automata. The label a abbreviates a set of labels (a, g), one for each g in G, and $(r,\, x_a = \perp \vee x_a \geq 5)$ stands for the set of labels (r, g') such that $g' \in G'$. The interval alphabet notation we use is certainly very unwieldy for describing automata, but it is convenient for the arguments we use in this paper.

3. A Logical Characterisation

The aim of this section is to characterise the class of ω-regular event clock languages via a monadic second order logic interpreted over timed words. We call the logic $\mathrm{MSO}_{ec}(\Sigma)$ and it is parameterised by the alphabet Σ.

Here and in the logics to follow, we assume a supply of individual variables $x, y, \ldots$, and set variables $X, Y, \ldots$. These variables will range over positions (respectively sets of positions) of a given timed word. We will make use of the predicates $Q_a(x)$, $\lhd_a(x) \in I$, and $\rhd_a(x) \in I$, where x is an individual variable, $a \in \Sigma$, and I is an element of $\mathcal{I}$. The syntax of $\mathrm{MSO}_{ec}(\Sigma)$ is given by:

$$\varphi ::= x \in X \mid x < y \mid Q_a(x) \mid \lhd_a(x) \in I \mid \rhd_a(x) \in I \mid \neg\varphi \mid (\varphi \vee \varphi) \mid \exists x \varphi \mid \exists X \varphi.$$

An interpretation for the logic will be a pair $(\sigma, \mathbb{I})$ where $\sigma \in T\Sigma^\omega$ and $\mathbb{I}$ is an assignment of individual variables to a position of σ (i.e. an element of $\mathbb{N}$), and set variables to a set of positions of σ. The predicate '$<$' is interpreted as the usual ordering on $\mathbb{N}$. The satisfaction relation $\sigma \models_{\mathbb{I}} \varphi$ for atomic formulas φ is given below. Let $\sigma = (\alpha, \eta)$. Then

$$\begin{array}{lll}
\sigma \models_{\mathbb{I}} x \in X & \text{iff} & \mathbb{I}(x) \in \mathbb{I}(X) \\
\sigma \models_{\mathbb{I}} x < y & \text{iff} & \mathbb{I}(x) < \mathbb{I}(y) \\
\sigma \models_{\mathbb{I}} Q_a(x) & \text{iff} & \alpha(\mathbb{I}(x)) = a \\
\sigma \models_{\mathbb{I}} \lhd_a(x) \in I & \text{iff} & v^{\sigma}_{\mathbb{I}(x)}(x_a) \in I \\
\sigma \models_{\mathbb{I}} \rhd_a(x) \in I & \text{iff} & v^{\sigma}_{\mathbb{I}(x)}(y_a) \in I.
\end{array}$$

The operator $\lhd_a(x)$ measures the time elapsed since the last a action w.r.t. the position x, and the predicate $\lhd_a(x) \in I$ asserts that this value lies in the interval I. The operator $\rhd_a(x)$ similarly refers to the time that will elapse before the *next* occurrence of a.

The operators $\neg$, $\vee$, and the existential quantifiers $\exists x$ and $\exists X$ are interpreted in the usual manner. In particular the quantifier $\exists X$ is interpreted as follows. Let $\mathbb{I}$ be an assignment for variables with respect to σ. Let $i \in \mathbb{N}$. We will use the notation $\mathbb{I}[i/x]$ to denote the assignment which maps x to i and agrees with $\mathbb{I}$ on all other individual and set variables. Similarly, for a subset S of $\mathbb{N}$, the notation $\mathbb{I}[S/X]$ will denote the interpretation which sends X to S, and agrees with $\mathbb{I}$ on all other variables. Then:

$$\sigma \models_{\mathbb{I}} \exists X \varphi \quad \text{iff} \quad \text{there exists } S \subseteq \mathbb{N} \text{ such that } \sigma \models_{\mathbb{I}[S/X]} \varphi.$$

Given a sentence φ in $\mathrm{MSO}_{ec}(\Sigma)$ we define $L(\varphi) = \{\sigma \in T\Sigma^\omega \mid \sigma \models \varphi\}$.

As an example, for $\Sigma' = \{a, r\}$ the following sentence ϕ in $\mathrm{MSO}_{ec}(\Sigma')$ describes the language accepted by the automaton of Fig. 2, namely that consecutive requests are separated by at least 5 time units:

$$\forall x (Q_r(x) \Rightarrow ((\lhd_r(x) = \bot) \vee (\lhd_r(x) \in [5, \infty))).$$

Theorem 1 *Let $L \subseteq T\Sigma^\omega$. Then L is an ω-regular event clock language over Σ iff $L = L(\varphi)$ for some sentence φ in* $\mathrm{MSO}_{ec}(\Sigma)$.

We will devote the rest of this section to the proof of this theorem. The proof will factor through the well-known logical characterisation of ω-regular languages due to Büchi [4] (see also [13]). Recall that for an alphabet A, Büchi's monadic second order logic (denoted here by $\mathrm{MSO}(A)$) is given as follows:

$$\varphi ::= x \in X \mid x < y \mid Q_a(x) \mid \neg\varphi \mid (\varphi \vee \varphi) \mid \exists x\varphi \mid \exists X\varphi.$$

An interpretation for this logic is a pair of the form $(\alpha, \mathbb{I})$ where $\alpha \in A^\omega$ and $\mathbb{I}$ assigns elements of $\mathbb{N}$ to individual variables, and subsets of $\mathbb{N}$ to set variables. The semantics of the logic is given in a similar manner to that of MSO_{ec}. In particular, the atomic formula $Q_a(x)$—here a is required to be in A—is interpreted as follows:

$$\alpha \models_{\mathbb{I}} Q_a(x) \text{ iff } \alpha(\mathbb{I}(x)) = a.$$

For a sentence φ in $\mathrm{MSO}(A)$ we set $L(\varphi) = \{\sigma \in A^\omega \mid \sigma \models \varphi\}$. Büchi's result then states that:

Theorem 2 ([4]) *A language $L \subseteq A^\omega$ is an ω-regular language over A iff $L = L(\varphi)$ for some sentence φ in* $\mathrm{MSO}(A)$.

Next, we introduce the notion of a *proper* interval alphabet which will play an important role in this paper. We say a finite set of intervals $\mathcal{S} \subseteq \mathcal{I}$ is *proper* if it forms a finite partition of $\mathbb{R}^{>0} \cup \{\perp\}$. Thus, if $\mathcal{S}$ is a proper interval set, then for each $t \in \mathbb{R}^{>0}$ there exists a unique $I \in \mathcal{S}$ such that $t \in I$, and for each $I, I' \in \mathcal{S}$, $I \neq I'$ implies $I \cap I' = \emptyset$. An interval alphabet Γ based on Σ will be termed proper if for each $z \in C_\Sigma$ the set $\Gamma_z = \{I \mid \exists(a,g) \in \Gamma \text{ with } g(z) = I\}$ is a proper interval set. We say an interval set $\mathcal{S}$ *covers* an interval set $\mathcal{S}'$ if every interval in $\mathcal{S}'$ is the union of some collection of intervals in $\mathcal{S}$. Finally, an interval alphabet Γ covers an interval alphabet Γ' (both based on Σ) if Γ_z covers Γ'_z for each $z \in C_\Sigma$.

Each interval alphabet Γ induces in a canonical way a proper interval alphabet, denoted $prop(\Gamma)$, with the property that it covers Γ. It is given by

$$prop(\Gamma) = \{(a,g) \mid a \in \Sigma \text{ and } \forall z \in C_\Sigma,\ g(z) \in prop(\Gamma_z)\}$$

where for each z, the set $prop(\Gamma_z)$ is obtained from Γ_z by the procedure outlined below.

Let $\mathcal{S}$ be a non-empty finite set of intervals. Let $B = \{r_0, r_1, \ldots, r_n\}$ be the set of bounds that appear in $\mathcal{S}$, and are different from $\perp$, 0, and ∞. If B is empty, we define $prop(\mathcal{S}) = \{[\perp,\perp], (0,\infty)\}$. Else, assuming without loss of generality that $r_0 < r_1 < \cdots < r_n$, we define

$$prop(\mathcal{S}) = \{[\perp,\perp], (0, r_0)\} \cup \{[r_i, r_i],\ (r_i, r_{i+1}) \mid 0 \leq i < n\} \cup \{[r_n, r_n], (r_n, \infty)\}.$$

It is easy to verify that $prop(\mathcal{S})$ is a proper interval set which covers $\mathcal{S}$.

Here is an important property of proper interval alphabets.

Lemma 1 *Let Γ be a proper interval alphabet based on Σ. Then for each $\sigma \in T\Sigma^\omega$ there exists a* unique *word $\gamma \in \Gamma^\omega$ such that $\sigma \in tw(\gamma)$.*

Proof. Let $\sigma \in T\Sigma^\omega$, with $\sigma = (\alpha, \eta)$, and let $i \in \mathbb{N}$. For each $z \in C_\Sigma$, there exists a *unique* interval I_i^z in Γ_z such that $v_i^\sigma(z) \in I_i^z$. This follows from the fact that Γ_z is a proper interval set, which in turn follows from Γ being a proper interval alphabet. Now define γ by $\gamma(i) = (\alpha(i), g_i)$, where $g_i(z) = I_i^z$ for each i. It follows that $\sigma \in tw(\gamma)$. The uniqueness of γ follows easily from the fact that there cannot be an interval $I \in \Gamma_z$ such that $I \neq I_i^z$ and $I \cap I_i^z \neq \emptyset$. □

Now, given a formula $\varphi \in \mathrm{MSO}_{ec}(\Sigma)$ we show how to translate it to a formula $t\text{-}s_\Gamma(\varphi) \in \mathrm{MSO}(\Gamma)$, for a suitably defined interval alphabet Γ. The translation will preserve, in a sense to be made precise in Lemma 2, the timed models of φ. The name t-s is meant as a mnemonic for "timed-to-symbolic". We denote by $voc_{x_a}(\varphi)$ (for "vocabulary"), the set of intervals I for which there exists a subformula of φ of the form $\lhd_a(x) \in I$, and similarly by $voc_{y_a}(\varphi)$ the set of intervals I for which there exists a subformula of φ of the form $\rhd_a(y) \in I$. Let Γ be any proper interval alphabet over Σ such that for each $z \in C_\Sigma$, Γ_z covers $voc_z(\varphi)$. Note that

$$\{(a, g) \mid a \in \Sigma, \text{ and } \forall z \in C_\Sigma,\ g(z) \in prop(voc_z(\varphi))\}$$

is at least one such Γ. Then $t\text{-}s_\Gamma(\varphi)$ is obtained from φ by replacing sub-formulas of the form $Q_a(x)$ by the formula

$$\bigvee_{(b,g)\in\Gamma,\ b=a} Q_{(b,g)}(x),$$

sub-formulas of the form $\lhd_a(x) \in I$ by the formula

$$\bigvee_{(b,g)\in\Gamma,\ g(x_a)\subseteq I} Q_{(b,g)}(x)$$

and $\rhd_a(x) \in I$ by

$$\bigvee_{(b,g)\in\Gamma,\ g(y_a)\subseteq I} Q_{(b,g)}(x).$$

Lemma 2 *Let $\varphi \in \mathrm{MSO}_{ec}(\Sigma)$ and let Γ be a proper interval alphabet based on Σ such that Γ_z covers $voc_z(\varphi)$ for each $z \in C_\Sigma$. Let $\gamma \in \Gamma^\omega$ and $\sigma \in T\Sigma^\omega$ be such that $\sigma \in tw(\gamma)$. Suppose further that $\mathbb{I}$ is an assignment for variables. Then*

1. *$\sigma \models_{\mathbb{I}} \varphi$ iff $\gamma \models_{\mathbb{I}} t\text{-}s_\Gamma(\varphi)$.*
2. *If φ is a sentence, then $L(\varphi) = tw(L(t\text{-}s_\Gamma(\varphi))$.*

Proof. *1.* We prove the statement by induction on the structure of φ. The interesting cases are $\varphi = Q_a(x)$ and $\varphi = \lhd_a(x) \in I$. Let $\sigma = (\alpha, \eta)$, and $\gamma = (a_0, g_0)(a_1, g_1)\cdots$.

Case $\varphi = Q_a(x)$: We know $\sigma \models_{\mathbb{I}} Q_a(x)$ iff $\alpha(\mathbb{I}(x)) = a$. But since $\sigma \in tw(\gamma)$, we know that this holds iff $\gamma(\mathbb{I}(x)) = (a, g)$ for some g such that $(a, g) \in \Gamma$. This in turn holds iff $\gamma \models_{\mathbb{I}} \bigvee_{(b,g')\in\Gamma,\ b=a} Q_{(b,g')}(x)$. Thus, $\sigma \models_{\mathbb{I}} \varphi$ iff $\gamma \models_{\mathbb{I}} t\text{-}s_\Gamma(\varphi)$.

Case $\varphi = \lhd_a\ x \in I$: Let $\sigma \models_{\mathbb{I}} \lhd_a\ x \in I$. Then we know that $v_{\mathbb{I}(x)}^\sigma(x_a) \in I$. Further, since $\sigma \in tw(\gamma)$, we know that

$$v_{\mathbb{I}(x)}^\sigma(x_a) \in g_{\mathbb{I}(x)}(x_a).$$

Using the fact that Γ_{x_a} is proper and covers $voc_{x_a}(\varphi)$, it must be the case that $g_{\mathbb{I}(x)}(x_a) \subseteq I$. Hence

$$\gamma \models_{\mathbb{I}} \bigvee_{(b,g)\in\Gamma, g(x_a)\subseteq I} Q_{(b,g)}(x).$$

Conversely, let

$$\gamma \models_{\mathbb{I}} \bigvee_{(b,g)\in\Gamma,\ g(x_a)\subseteq I} Q_{(b,g)}(x).$$

Then $\gamma(\mathbb{I}(x)) = (b, g)$ for some $(b, g) \in \Gamma$ such that $g(x_a) \subseteq I$. That is, $g_{\mathbb{I}(x)}(x_a) \subseteq I$. Now since $\sigma \in tw(\gamma)$ we have $v^{\sigma}_{\mathbb{I}(x)}(x_a) \in g_{\mathbb{I}(x)}(x_a)$. Since $g_{\mathbb{I}(x)}(x_a) \subseteq I$, we have $v^{\sigma}_{\mathbb{I}(x)}(x_a) \in I$, and hence $\sigma \models_{\mathbb{I}} \lhd_a(x) \in I$.

2. This is easy to see once we have *(1)* above. Let $\sigma \models \varphi$. Then, using Lemma 1, there exists a $\gamma \in \Gamma^{\omega}$ such that $\sigma \in tw(\gamma)$. Using *(1)* above, we have $\gamma \in L(t\text{-}s_{\Gamma}(\varphi))$ and hence $\sigma \in tw(L(t\text{-}s_{\Gamma}(\varphi)))$. Conversely, if $\sigma \in tw(\gamma)$ and $\gamma \models t\text{-}s_{\Gamma}(\varphi)$, then by *(1)* again we have that $\sigma \models \varphi$, and hence $\sigma \in L(\varphi)$. $\square$

Let Γ be a proper interval alphabet based on Σ. We now show how we can associate a formula $s\text{-}t(\widehat{\varphi}) \in \mathrm{MSO}_{ec}(\Sigma)$ with a formula $\widehat{\varphi} \in \mathrm{MSO}(\Gamma)$, such that, once again, the translated formula preserves timed models. The formula $s\text{-}t(\widehat{\varphi})$ is obtained by replacing atomic sub-formulas in $\widehat{\varphi}$ of the form $Q_{(a,g)}(x)$ by the formula

$$Q_a(x) \wedge \bigwedge_{x_b\in C_\Sigma} \lhd_b(x) \in g(x_b) \ \wedge \bigwedge_{y_b\in C_\Sigma} \rhd_b(x) \in g(y_b).$$

The following lemma is easy to show along the lines of Lemma 2:
Lemma 3 *Let* Γ *be a proper interval alphabet based on* Σ *and let* $\widehat{\varphi}$ *be a sentence in* $\mathrm{MSO}(\Gamma)$*. Then* $L(s\text{-}t(\widehat{\varphi})) = tw(L(\widehat{\varphi}))$. $\square$

We are now in a position to provide a proof of Theorem 1. Let L be an ω-regular event clock language over Σ. It is not difficult to see that there must be a *proper* interval alphabet Γ based on Σ, and an ω-regular subset $\widehat{L}$ of Γ^{ω} such that $L = tw(\widehat{L})$. Büchi's theorem tells us that there exists an $\mathrm{MSO}(\Gamma)$-sentence $\widehat{\varphi}$ such that $L(\widehat{\varphi}) = \widehat{L}$. Hence $L = tw(L(\widehat{\varphi}))$. By Lemma 3, we have a $\mathrm{MSO}_{ec}(\Sigma)$-sentence, namely $\varphi = s\text{-}t(\widehat{\varphi})$ (w.r.t. Γ), such that $L(\varphi) = tw(L(\widehat{\varphi}))$. Thus $L = L(\varphi)$.

Conversely, let φ be a $\mathrm{MSO}_{ec}(\Sigma)$-sentence. Let Γ be a proper interval alphabet based on Σ such that Γ_z covers $voc_z(\varphi)$ for each $z \in C_{\Sigma}$. Then, by Lemma 2, we know that there exists a formula $\widehat{\varphi} = t\text{-}s_{\Gamma}(\varphi)$ in $\mathrm{MSO}(\Gamma)$, such that $L(\varphi) = tw(L(\widehat{\varphi}))$. Using Büchi's theorem once more, we know that $L(\widehat{\varphi})$ is an ω-regular language over Γ. Thus $L(\varphi)$ is an ω-regular event clock language over Σ. This completes the proof of Theorem 1. $\square$

The proof of Theorem 1 gives us a way to decide the satisfiability problem for the logic MSO_{ec}. The translation used are all effective (including the one from MSO to Büchi automata). Once we have an event clock automaton for the given MSO_{ec} sentence, we know that its language is empty iff the given MSO_{ec} sentence is unsatisfiable. The language emptiness problem for event clock automata can be decided using either a direct region construction or a translation to timed automata, as done in [2].

4. Projection and Existential Quantification in MSO_{ec}

In this section we point out the nature of the projection operation associated with existential quantification in the logic MSO_{ec}. In classical monadic logics (as in S1S) existential quantification usually corresponds to the operation of projection (a special form of renaming) on the languages corresponding to formulas in the logic. In S1S for example existential quantification corresponds to projection on ω-regular languages, and the class of ω-regular languages are closed under projections (in fact under general renamings). In the case of MSO_{ec} however, existential quantification corresponds to closure under renaming of a subclass of event clock languages which we call *quasi* event clock languages. We will formalise these ideas below.

Let U be a (possibly infinite) universe of letters (from which our alphabets will be drawn). Consider a finite partition of this universe, given by a function f from the universe U to a finite indexing set C.

Let us fix such a triple (U, f, C). Let $A \subseteq U$ be an alphabet of actions. Then a *quasi* event clock automaton (QECA) over A (w.r.t. the triple (U, f, C)) is similar to an event clock automaton over A except that the clock x_a records the time since the last action b in A such that $f(b) = f(a)$. Similarly, the clock y_a records the time to the next action b such that $f(b) = f(a)$. We will say $L \subseteq TU^\omega$ is an (ω-regular) *quasi event clock language* (QECL) w.r.t (U, f, C) if $L = L(\mathcal{A})$ for some alphabet $A \subseteq U$, and QECA $\mathcal{A}$ over A.

The class of QECL's w.r.t. (U, f, C) can, in general, be seen to be a strict subclass of the class of ω-regular event clock languages over alphabets which are subsets of U. It is not difficult to see that an event clock automaton over A can simulate a QECA over A. Further, if a block of f contains two or more symbols, say a and b, then the language

$$L = \{(a, t_0)(b, t_1)(a, t_2) \mid t_2 = t_0 + 1\}$$

is an ω-regular event clock language over $\{a, b\}$, but there is *no* corresponding QECL over $\{a, b\}$, w.r.t. (U, f, C).

Let A, A' be alphabets, with $A, A' \subseteq U$. A renaming from A to A' is a map from A to A'. We say $\varsigma : A \to A'$ is a *valid* renaming w.r.t. (U, f, C) iff for each $a \in A$, $f(a) = f(\varsigma(a))$ (i.e. both $\varsigma(a)$ and a belong to the same block of the partition). For a timed word $\sigma \in TA^\omega$ and a timed language $L \subseteq TA^\omega$, the timed word $\varsigma(\sigma) \in TA'^\omega$ and timed language $\varsigma(L) \subseteq TA'^\omega$ are defined in the expected manner. The following proposition is then easily verified:

Proposition 3 *The class of quasi event clock languages over (U, f, C) is closed under renaming operations which are valid w.r.t. (U, f, C).* $\square$

If we now consider a direct proof of Theorem 1 along the lines of the one for MSO (see [13]), it will be clear that existential quantification in MSO_{ec} corresponds to the restricted renaming defined above, on the class of QECL's. We will concentrate on one direction of the proof where we show that every $\mathrm{MSO}_{ec}(\Sigma)$ sentence can be captured by an event clock automaton over Σ. This is done by associating, inductively, a QECA with each formula in $\mathrm{MSO}_{ec}(\Sigma)$.

Let $\varphi(m,n)$ denote a formula whose free variables are among

$$\{x_1,\ldots,x_m,X_1,\ldots,X_n\}.$$

An interpretation $(\sigma,\mathbb{I})$ for a formula $\varphi(m,n)$ in $\mathrm{MSO}_{ec}(\Sigma)$ is encoded as a timed word over Σ_{m+n} where for $i\geq 0$, Σ_i is defined to be

$$\Sigma\times\overbrace{\{0,1\}\times\cdots\times\{0,1\}}^{i}.$$

Let $\sigma=(\alpha,\eta)$. Then $(\sigma,\mathbb{I})$ is represented as $\sigma'\in T(\Sigma_{m+n})^\omega$ with $\sigma'=(\alpha',\eta)$ where α' is given as follows: for each $i\in\mathbb{N}$, $\alpha'(i)=(a,b_1,\ldots b_m,c_1,\ldots,c_n)$ where $a=\alpha(i)$, $b_j=1$ iff $\mathbb{I}(x_j)=i$ and $c_j=1$ iff $i\in\mathbb{I}(X_j)$. The satisfaction relation $\sigma'\models\varphi(m,n)$, is defined in the expected manner based on the semantics given earlier. Each formula $\varphi(m,n)$ in $\mathrm{MSO}_{ec}(\Sigma)$ thus defines a subset of $T(\Sigma_{m+n})^\omega$, denoted $L(\varphi(m,n))$, which is the set of models of φ.

In the induction step for the "$\exists X$" case, we assume that the formula $\varphi(m,n+1)$ is such that $L(\varphi(m,n+1))\subseteq T\Sigma^\omega_{m+n+1}$ is accepted by a QECA over Σ_{m+n+1}, with respect to the triple (U,h,Σ) where h (restricted to $\bigcup_{i\geq 0}\Sigma_i$) is given by $h((a,d_1,\ldots,d_l))=a$. Then the set of models of the formula $(\exists X_{n+1}\varphi)(m,n)$ is simply obtained from $L(\varphi(m,n+1))$ by projecting each letter of Σ_{m+n+1} to the dimensions 1 to $m+n$. This projection is clearly a valid renaming w.r.t. (U,h,Σ), and by Proposition 3 the language $L((\exists X_{n+1}\varphi)(m,n))$ is also a QECL over Σ_{m+n}, w.r.t. (U,h,Σ). Note that in this way, for a sentence φ, the language $L(\varphi)$ is accepted by a QECA over Σ (w.r.t. (U,h,Σ)), which is nothing but an event clock automaton over Σ.

Thus, the existential quantification in $\mathrm{MSO}_{ec}(\Sigma)$ corresponds to a renaming which is valid w.r.t. to (U,h,Σ), applied on quasi event clock languages w.r.t. (U,h,Σ).

5. Expressive Completeness of LTL_{ec}

In this section we formulate a version of the timed temporal logic nrEventClockTL introduced in [11] and called here LTL_{ec}. We show that LTL_{ec} is expressively equivalent to the first-order fragment of MSO_{ec}.

The formulas of $\mathrm{LTL}_{ec}(\Sigma)$ (parameterised by the alphabet Σ) are given by:

$$\varphi ::= \top \mid a \mid \;\lhd_a\in I \mid \;\rhd_a\in I \mid O\varphi \mid \neg\varphi \mid (\varphi\vee\varphi) \mid (\varphi U\varphi).$$

Here we require $a\in\Sigma$.

The models for $\mathrm{LTL}_{ec}(\Sigma)$ formulas are timed words over Σ. Let $\sigma\in T\Sigma^\omega$, with

$\sigma = (\alpha, \eta)$, and let $i \in \mathbb{N}$. Then the satisfaction relation $\sigma, i \models \varphi$ is given by

$$\begin{array}{lll}
\sigma, i \models \top & & \\
\sigma, i \models a & \text{iff} & \alpha(i) = a \\
\sigma, i \models \lhd_a \in I & \text{iff} & v_i^\sigma(x_a) \in I \\
\sigma, i \models \rhd_a \in I & \text{iff} & v_i^\sigma(y_a) \in I \\
\sigma, i \models O\varphi & \text{iff} & \sigma, i+1 \models \varphi \\
\sigma, i \models \neg\varphi & \text{iff} & \sigma, i \not\models \varphi \\
\sigma, i \models \varphi \vee \varphi' & \text{iff} & \sigma, i \models \varphi \text{ or } \sigma, i \models \varphi' \\
\sigma, i \models \varphi U \varphi' & \text{iff} & \exists k \geq i : \sigma, k \models \varphi' \text{ and } \forall j : i \leq j < k,\ \sigma, j \models \varphi.
\end{array}$$

We say $\sigma \models \varphi$ iff $\sigma, 0 \models \varphi$. Define $L(\varphi) = \{\sigma \in T\Sigma^\omega \mid \sigma \models \varphi\}$.

As an example, the $\mathrm{LTL}_{ec}(\Sigma)$ formula

$$\Box(r \Rightarrow (\lhd_r \in [\bot, \bot]) \vee (\lhd_r \in [5, \infty))),$$

where $\Box\alpha$ is an abbreviation for $\neg(\top U \neg\alpha)$, rephrases the property ϕ of Sec. 3.

Let $\mathrm{FO}_{ec}(\Sigma)$ denote the first-order fragment of the logic $\mathrm{MSO}_{ec}(\Sigma)$. Thus $\mathrm{FO}_{ec}(\Sigma)$ is obtained from $\mathrm{MSO}_{ec}(\Sigma)$ by disallowing the use of quantification over set variables. The aim of the rest of this section is to prove the following result.

Theorem 4 $\mathrm{LTL}_{ec}(\Sigma)$ *is expressively equivalent to* $\mathrm{FO}_{ec}(\Sigma)$.

The method of proof will be to translate LTL_{ec} formulas into classical LTL over an appropriate interval alphabet. The method is similar to the proof of Theorem 1 and we will also make use of the translation used there.

It will be useful to first recall the definition of LTL and the result concerning its expressive completeness. Let A be an alphabet of actions. Then the formulas of "action-based" LTL, denoted $\mathrm{LTL}(A)$, are given by the syntax:

$$\varphi ::= \top \mid a \mid O\varphi \mid \neg\varphi \mid (\varphi \vee \varphi) \mid (\varphi U \varphi).$$

Once again, we require $a \in A$. The semantics of $\mathrm{LTL}(A)$ is given similarly to $\mathrm{LTL}_{ec}(\Sigma)$ above, with models being infinite words over A. In particular, for a word $\alpha \in A^\omega$ we have

$$\alpha, i \models a \quad \text{iff} \quad \alpha(i) = a.$$

We will say that $\alpha \models \varphi$ iff $\alpha, 0 \models \varphi$. We set $L_{sym}(\varphi) = \{\alpha \in A^\omega \mid \alpha \models \varphi\}$.

Let $\mathrm{FO}(A)$ denote the first-order fragment of the logic $\mathrm{MSO}(A)$. As before, $\mathrm{FO}(A)$ is obtained from the logic $\mathrm{MSO}(A)$ defined in Section 3, by disallowing the use of set variables. Then a well known result due to the work of Kamp, and Gabbay et. al (see also [15]), is:

Theorem 5 ([7, 8]) $\mathrm{LTL}(A)$ *is expressively equivalent to* $\mathrm{FO}(A)$. $\Box$

In manner analogous to MSO_{ec} we define translations between the timed and symbolic versions of LTL. By abuse of notation we denote these translations also by *s-t* and *t-s*.

Let φ be a $\mathrm{LTL}_{ec}(\Sigma)$ formula. Let Γ be a proper interval alphabet based on Σ, such that Γ_z covers $voc_z(\varphi)$ for each $z \in C_\Sigma$ (we extend the definition of $voc_z(\varphi)$

to LTL_{ec} formulas in the natural way). Then the formula $t\text{-}s_\Gamma(\varphi)$ in $\mathrm{LTL}(\Gamma)$ is obtained by replacing subformulas of the form a by

$$\bigvee_{(b,g)\in\Gamma, b=a} (b,g),$$

$\lhd_a \in I$ by

$$\bigvee_{(b,g)\in\Gamma, g(x_a)\subseteq I} (b,g),$$

and $\rhd_a \in I$ by

$$\bigvee_{(b,g)\in\Gamma, g(y_a)\subseteq I} (b,g).$$

The translation $s\text{-}t(\widehat{\varphi})$ for a formula $\widehat{\varphi}$ in $\mathrm{LTL}(\Gamma)$ is defined in the expected manner, as done in Sec. 3.

The relationship between the translated formulas is summarised below:

Lemma 4 *Let Γ be a proper interval alphabet based on Σ. Then*

1. *if φ is an $\mathrm{LTL}_{ec}(\Sigma)$ formula such that Γ_z covers $voc_z(\varphi)$ for each $z \in C_\Sigma$, then $L(\varphi) = tw(L(t\text{-}s_\Gamma(\varphi)))$.*
2. *if $\widehat{\varphi}$ is a formula in $\mathrm{LTL}(\Gamma)$, then $L(s\text{-}t(\widehat{\varphi})) = tw(L(\widehat{\varphi}))$.*

Proof. The proof of this lemma is very similar to the proof of Lemma 2 which makes use of the properties of proper interval sets. □

Returning now to the proof of Theorem 4, let $\varphi \in \mathrm{LTL}_{ec}(\Sigma)$. Then once again, we can construct a proper interval alphabet Γ based on Σ such that Γ_z covers $voc_z(\varphi)$ for each $z \in C_\Sigma$. Then from Lemma 4 we know that the formula $\widehat{\varphi} = t\text{-}s_\Gamma(\varphi)$ in $\mathrm{LTL}(\Gamma)$ is such that $tw(L(\widehat{\varphi})) = L(\varphi)$. Now, by Theorem 5, we know that there exists a sentence $\widehat{\varphi}_1$ in $\mathrm{FO}(\Gamma)$ such that $L(\widehat{\varphi}_1) = L(\widehat{\varphi})$. Now consider the sentence $\varphi_1 = s\text{-}t(\widehat{\varphi}_1)$ w.r.t. the proper interval alphabet Γ (cf. Sec. 3). The translations $s\text{-}t$ and $t\text{-}s$ are such that if the given formula is first-order, then so is the translated formula. Thus φ_1 is a $\mathrm{FO}_{ec}(\Sigma)$ sentence. Further, since Γ is proper, by Lemma 3 we know that $L(\varphi_1) = tw(L(\widehat{\varphi}_1))$. Thus φ_1 is the required $\mathrm{FO}_{ec}(\Sigma)$ sentence with $L(\varphi) = L(\varphi_1)$.

Conversely, let φ_1 be a sentence in $\mathrm{FO}_{ec}(\Sigma)$. Then, once again, there exists a proper interval alphabet Γ based on A such that Γ_a covers $voc_z(\varphi_1)$ for each $z \in C_\Sigma$. Consider the $\mathrm{MSO}(\Gamma)$ sentence $\widehat{\varphi}_1 = t\text{-}s_\Gamma(\varphi_1)$ with respect to the interval alphabet Γ (cf. Section 3). By Lemma 2, $tw(L(\widehat{\varphi}_1)) = tw(L(\varphi_1))$. Further, $\widehat{\varphi}_1$ is a sentence in $\mathrm{FO}(\Gamma)$. Now, again appealing to Theorem 5, we know that there exists an $\mathrm{LTL}(\Gamma)$ formula $\widehat{\varphi}$ such that $L(\widehat{\varphi}) = L(\widehat{\varphi}_1)$. By Lemma 4, we know that $L(\widehat{\varphi}) = tw(L(\varphi))$, where $\varphi = s\text{-}t(\widehat{\varphi})$. Thus φ is the required formula in $\mathrm{LTL}_{ec}(\Sigma)$ such that $L(\varphi) = L(\varphi_1)$. □

6. Conclusion

The class of event clock automata are a useful class of timed automata from the point of view of modelling and specifying real-time behaviours. They have important closure properties and we show in this paper that they also admit a clean logical characterisation as well as an expressively complete and natural timed temporal logic.

It would be interesting to know if the classical characterisations in terms of algebra and regular expressions also extend to this class of automata. It would also be interesting to investigate the differences between the pointwise semantics used in this paper and the continuous semantics used in [9], and whether a similar logical characterisation exists in the case of continuous semantics.

Acknowledgments

I thank P. S. Thiagarajan for several useful inputs.

References

1. R. Alur, D. L. Dill: A theory of timed automata, *Theoretical Computer Science* **126**: 183–235 (1994).
2. R. Alur, L. Fix, T. A. Henzinger: Event-clock automata: a determinizable class of timed automata, *Proc. 6th Int. Conf. on Computer-aided Verification,* LNCS **818**, 1–13, Springer-Verlag (1994).
3. R. Alur, T. A. Henzinger: Real-time logics: complexity and expressiveness, *Information and Computation* **104**, 35–77 (1993).
4. J. R. Büchi: Weak second-order arithmetic and finite automata, *Zeitschrift für Math. Logik und Grundlagen der Mathematik*, **6**, 66–92 (1960).
5. D. D'Souza: A Logical Characterisation of Event Recording Automata, in *Proc. Int. Sym. on Formal Methods in Real-Time and Fault Tolerance (FTRTFT)* 2000, LNCS **1926** (2000).
6. D. D'Souza, P. S. Thiagarajan: Product Interval Automata: A Subclass of Timed Automata, *Proc. 19th Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, LNCS **1732** (1999).
7. A. W. H. Kamp: Tense Logic and the Theory of Linear Order, PhD Thesis, University of California (1968).
8. D. Gabbay, A. Pnueli, S. Shelah, J. Stavi: The Temporal Analysis of Fairness, *Seventh ACM Sym. on Principles of Programming Languages,* 163–173 (1980).
9. T. A. Henzinger, J.-F. Raskin, and P.-Y. Schobbens: The regular real-time languages, *Proc. 25th Int. Col. on Automata, Languages, and Programming 1998*, LNCS **1443**, 580–591 (1998).
10. A. Pnueli: The temporal logic of programs, *Proc. 18th IEEE Sym. on Foundation of Computer Science*, 46–57 (1977).
11. J. -F. Raskin: Logics, Automata and Classical Theories for Deciding Real Time, Ph.D Thesis, FUNDP, Belgium.
12. J. -F. Raskin, P. -Y. Schobbens: State-clock Logic: A Decidable Real-Time Logic, *Proc. HART '97: Hybrid and Real-Time Systems*, LNCS **1201**, 33–47 (1997).
13. W. Thomas: Automata on Infinite Objects, in J. V. Leeuwen (Ed.), *Handbook of*

Theoretical Computer Science, Vol. B, 133–191, Elsevier Science Publ., Amsterdam (1990).

14. Th. Wilke: Specifying Timed State Sequences in Powerful Decidable Logics and Timed Automata, in *Proc. Formal Techniques in Real-Time and Fault-Tolerant Systems*, LNCS **863**, 694–715 (1994).
15. Th. Wilke: Classifying Discrete Temporal Properties, in *Proc. Sym. Theoretical Aspects of Computer Science (STACS) 1999*, LNCS **1563**, 32–46 (1999).