Software Tools for Technology Transfer manuscript No.
(will be inserted by the editor)

On the expressiveness of MTL in the pointwise and continuous semantics

Deepak D'Souza, Pavithra Prabhakar

Department of Computer Science and Automation, Indian Institute of Science, Bangalore 560012, India.
Email: {deepakd, pavithra}@csa.iisc.ernet.in

Received: date / Revised version: date

Abstract. We show that the pointwise version of the logic MTL is strictly less expressive than the continuous version, over *finite* words. The proof is constructive in that we exhibit a timed language which is definable in the continuous semantics but is not definable in the pointwise semantics.

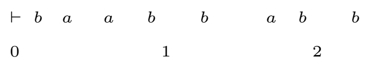
1 Introduction

The timed temporal logic *Metric Temporal Logic* (MTL) [8] has received much attention in the literature on the verification of real-time systems. MTL extends the until operator of classical temporal logic with an interval which specifies the time interval within which the formula must be satisfied. Over dense time the logic has traditionally been interpreted in two ways which have come to be known as the "pointwise" and the "continuous" semantics. In the pointwise version temporal assertions are interpreted only at time points where an "action" or "event" happens in the observed timed behaviour of a system, whereas in the continuous version one is allowed to assert formulas at arbitrary time points between events as well. For instance in the timed word containing exactly two events, an a at time 1 and a b at time 3, the MTL formula $\Diamond_{[1,1]}b$ ("a b occurs at a distance of 1 time unit") is not true at any point in this model in the pointwise semantics, whereas it is true at the time instant 2 in the continuous semantics.

There have been a variety of results regarding decision procedures and to an extent expressiveness of the logic in these two semantics. Alur, Feder, and Henzinger [3, 2] argue that the logic is undecidable in the continuous semantics, though a restriction to non-singular intervals (called "Metric Interval Temporal Logic", or MITL) is shown to be decidable. These results hold for both finite and infinite models. Recently Ouaknine and Worrell [11] have shown that MTL over finite words in the pointwise semantics is decidable, using a translation to one-clock alternating timed automata (ATA) which in turn were recently introduced and shown to be decidable ([9, 11]). There have also been a few expressiveness results, namely expressive completeness with respect to a class of "input-determined" timed automata in pointwise semantics [6], and expressive equivalence of MITL with an event-clock logic in continuous semantics [7].

The relative expressiveness of the pointwise and continuous semantics was however not addressed in the literature until the work in this paper and subsequently [4]. One expects the continuous version to be at least as expressive as the pointwise one since we can characterise the action points in the continuous semantics, and hence mimic the pointwise semantics by restricting the interpretation to only these points. Intuitively, one would also expect the continuous version to be strictly more expressive, given its ability to quantify over arbitrary time points.

In this paper we confirm this intuition for the case of finite models by exhibiting a language of finite timed words which is definable in the continuous semantics but *not* in the pointwise semantics. The language we consider is L_{ni} (for "no insertions") over the alphabet $\{a, b\}$, which consists of timed words in which for any two consecutive a's the time period between them translated by one time unit contains no events.



This language can easily be seen to be expressible in the continuous semantics. However, we argue that it is not expressible in the pointwise semantics, since it would essentially lead to the undecidability of the pointwise version of the logic, which would contradict the result of [11].

It is appropriate to point out here that the fact that one version of the logic is decidable while the other isn't, does not necessarily mean that one is less expressive than the other. It is crucial in our argument that we are able to identify a property which is in a sense *independent* of the problem instance being translated. In fact, such a property may not exist in general. This is borne out by the fact that there are logics [10] with two equally expressive versions, one of which is decidable and the other isn't.

In the more recent and independent work of [4], the authors prove a variety of results concerning the expressiveness of the logics MTL and TPTL in the pointwise and continuous semantics, including a result similar to ours for infinite models. They use a more classical argument by identifying two parameterised families of models which a continuous formula can distinguish, but no pointwise one can. In contrast our result is for finite models and uses a simple though novel argument which appears useful in obtaining similar inexpressiveness results in the timed setting. For instance, in the extended version of this paper [5], we use a similar argument to show that 1-clock ATA's cannot express the complement of the language L_{ni}, while timed automata can.

2 Syntax and semantics of MTL

As usual, A^* will denote the set of finite words over an alphabet A. For a finite word $w = a_0 a_1 \cdots a_n$, we use $|w|$ to denote the length of w (in this case $n+1$) and $w(i)$ to denote a_i, the i-th symbol of w.

To avoid trivial differences in expressiveness, we define timed words with a special left-end marker '$\vdash$'. A (finite) *timed word* σ over an alphabet Σ is an element of $((\Sigma \cup \{\vdash\}) \times \mathbb{R}_{\geq 0})^*$ of the form $(\vdash, 0)(a_1, t_1) \cdots (a_n, t_n)$, satisfying $a_i \in \Sigma$, and for all $i \in \{1, \ldots, n-1\}$, $0 < t_i < t_{i+1}$. Wherever convenient we will also represent a timed word $\sigma = (a_0, t_0)(a_1, t_1) \cdots (a_n, t_n)$ as a pair $\sigma = (w, \tau)$, where $w = a_0 a_1 \cdots a_n$, and $\tau = t_0 t_1 \cdots t_n$. We use $\mathit{time}(\sigma)$ to denote the time of the last action, namely t_n. We write $T\Sigma^*$ for the set of timed words over Σ.

We now give the syntax and semantics of the two versions of MTL. The models for both versions will be timed words. For ease of comparison we also use a common syntax for the two logics, using an until operator U^s which is "strict" in its first argument. While the operator is the standard one used in the continuous version, the more natural operators of the pointwise semantics ("next" and a non-strict "until") continue to be expressible using this until operator [5].

Let us fix an alphabet Σ for the rest of this section. The formulas of MTL over Σ are defined as follows:

$$\varphi := a \mid \neg\varphi \mid (\varphi \vee \varphi) \mid (\varphi\ U_I^s\ \varphi),$$

where $a \in (\Sigma \cup \{\vdash\})$ and I is an interval with rational end points or ∞.

We first define the *pointwise* semantics for MTL. Given an MTL formula φ, a timed word $\sigma = (w, \tau)$, and a position i in the timed word (i.e. $0 \leq i < |\sigma|$), the satisfaction relation $\sigma, i \models_{pw} \varphi$ (read "σ at position i satisfies φ in the pointwise semantics") is inductively defined as:

$$\begin{array}{ll}
\sigma, i \models_{pw} a & \text{iff } w(i) = a. \\
\sigma, i \models_{pw} \neg\varphi & \text{iff } \sigma, i \not\models_{pw} \varphi. \\
\sigma, i \models_{pw} \varphi_1 \vee \varphi_2 & \text{iff } \sigma, i \models_{pw} \varphi_1 \text{ or } \sigma, i \models_{pw} \varphi_2. \\
\sigma, i \models_{pw} \varphi_1 U_I^s \varphi_2 & \text{iff } \exists j \geq i \colon \tau(j) - \tau(i) \in I,\ \sigma, j \models_{pw} \varphi_2, \\
& \quad \text{and } \forall k : i < k < j,\ \sigma, k \models_{pw} \varphi_1.
\end{array}$$

We use MTL^{pw} to denote the pointwise interpretation of this logic. The language defined by an MTL^{pw} formula φ in pointwise semantics is given by $L^{pw}(\varphi) = \{\sigma \in T\Sigma^* \mid \sigma, 0 \models_{pw} \varphi\}$. The derived operators $\Diamond_I$ and $\Box_I$ are defined with the expected meaning: $\Diamond_I \varphi$ is $\top U_I^s \varphi$ and $\Box_I \varphi$ is $\neg\Diamond_I \neg\varphi$. The standard untimed LTL operators of O, $\Diamond$ and $\Box$ can also be defined via $(\neg\top) U^s_{(0,\infty)} \varphi$, $\Diamond_{[0,\infty)}$ and $\Box_{[0,\infty)}$ respectively.

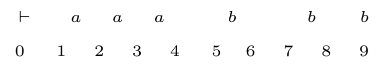
The continuous version of MTL has typically been interpreted over timed state sequences (e.g. in [2]), where a system is assumed to continue in a state till the next transition or state change takes place. Here we interpret the logic over timed words, while keeping intact the essence of the continuous interpretation, namely the ability to assert formulas at arbitrary time points. This gives us a common class of models to compare the two semantics. In our interpretation the atomic formula a can only hold at an action point labelled a, and not during an interval of time after the event happens. This view is appropriate given that we are talking about event occurrences and not propositions that make up a state.

Given an MTL formula φ, a timed word $\sigma = (w, \tau)$, and a time t such that $0 \leq t \leq \mathit{time}(\sigma)$, the satisfaction relation $\sigma, t \models_c \varphi$ (read "σ at time t satisfies φ in the continuous semantics") is inductively defined as follows:

$$\begin{array}{ll}
\sigma, t \models_c a & \text{iff } \exists i \colon \tau(i) = t \text{ and } w(i) = a. \\
\sigma, t \models_c \neg\varphi & \text{iff } \sigma, t \not\models_c \varphi. \\
\sigma, t \models_c \varphi_1 \vee \varphi_2 & \text{iff } \sigma, t \models_c \varphi_1 \text{ or } \sigma, t \models_c \varphi_2. \\
\sigma, t \models_c \varphi_1 U_I^s \varphi_2 & \text{iff } \exists t' \colon t' \geq t, t' - t \in I,\ \sigma, t' \models_c \varphi_2 \\
& \quad \text{and } \forall t'' : t < t'' < t',\ \sigma, t'' \models_c \varphi_1.
\end{array}$$

We use MTL^c to denote this interpretation of MTL, and define the language corresponding to an MTL^c formula φ as $L^c(\varphi) = \{\sigma \in T\Sigma^* \mid \sigma, 0 \models_c \varphi\}$.

To illustrate the two interpretations, consider the timed word $(\vdash, 0)(a, 1.3)(a, 2.5)(a, 3.6)(b, 5.3)(b, 7.6)(b, 9)$ depicted below.



The formula $\Diamond_{[4,5]}(\Diamond_{[1,1]} b)$ is satisfied by the above timed word in the continuous semantics, since we have a point in the interval [4,5] such that there is a b at distance one from this point. However, the timed word does not satisfy this formula in the pointwise semantics as there is no *action point* in the interval [4,5] (and hence no action point which satisfies $(\Diamond_{[1,1]}b)$).

3 Strict containment of MTL^{pw} in MTL^{c}

In this section we show that MTL^{pw} is strictly less expressive than MTL^{c}. It is not difficult to see that an MTL^{pw} formula φ can be translated to an equivalent MTL^{c} formula $pw\text{-}c(\varphi)$ by forcing formulas to be interpreted at action points only. In the translation below, the formula $\varphi_{act} = \bigvee_{a\in(\Sigma\cup\{\vdash\})} a$ describes the action points in a timed word.

$$\begin{array}{ll} pw\text{-}c(a) & = a \text{ where } a \in (\Sigma \cup \{\vdash\}). \\ pw\text{-}c(\neg\varphi) & = \neg pw\text{-}c(\varphi). \\ pw\text{-}c(\varphi \vee \psi) & = pw\text{-}c(\varphi) \vee pw\text{-}c(\psi). \\ pw\text{-}c(\varphi U^s_I \psi) & = (\neg\varphi_{act} \vee pw\text{-}c(\varphi)) U^s_I (\varphi_{act} \wedge pw\text{-}c(\psi)). \end{array}$$

It is easy to see that $L^{pw}(\varphi) = L^c(pw\text{-}c(\varphi))$, and hence that MTL^{pw} is no more expressive than MTL^{c}.

To argue strict containment of MTL^{pw} in MTL^{c}, we argue that the language L_{ni} over the alphabet $\{a, b\}$, which is defined by the MTL^{c} formula

$$\neg\Diamond(a \wedge (\neg\varphi_{act}U^s_{(0,\infty)}a) \wedge (\neg a U^s_{(0,\infty)}(\neg a \wedge \Diamond_{[1,1]}(a \vee b)))),$$

is not definable by any MTL formula in the pointwise semantics. To give a language-theoretic definition of L_{ni} let us first define the language $L^{\Sigma,c}_{ni}$ which is parameterised by the alphabet Σ and an action $c \in \Sigma$. The language $L^{\Sigma,c}_{ni}$ consists of timed words $\sigma \in T\Sigma^*$ of the form $(a_0, t_0)\cdots(a_n, t_n)$, satisfying the condition that whenever $a_i = a_{i+1} = c$ for some i, for no j does t_j belong to the interval $(t_i + 1, t_{i+1} + 1)$. The language L_{ni} can now be defined as $L^{\{a,b\},a}_{ni}$.

Suppose there existed an MTL^{pw} formula φ_{ni} which described the language L_{ni}. Then one can verify that the formula $\varphi_{ni}[c/a, (\bigvee_{d\in\Sigma-\{c\}} d)/b]$ obtained from φ_{ni} by replacing every occurrence of a by c, and b by $\bigvee_{d\in\Sigma-\{c\}} d$, defines the language $L^{\Sigma,c}_{ni}$. Let us call this formula $\varphi^{\Sigma,c}_{ni}$.

We now argue that the assumption that the MTL^{pw} formula φ_{ni} exists leads to the undecidability of the satisfiability problem for MTL^{pw}. Assuming that φ_{ni} exists, we give a reduction from the halting problem of a 2-counter machine on an empty tape. The reduction is along the lines of [3], [1] and [2], though the encoding is adapted to enable us to isolate the language above.

Recall that a 2-counter machine M has 2 counters C and D and a sequence of n instructions, the last of which is a halt instruction. The rest of the instructions are increment (or decrement) of a counter together with a jump to another location, or a jump conditional on the value of a counter being 0. A configuration of M is represented by a triple $\langle i, c, d\rangle$, where $i \in \{1, \ldots, n\}$ is the location counter, and c and d are the values of the two counters C and D respectively. A halting computation of M is a finite sequence of configurations, starting with the configuration $< 1, 0, 0 >$, where successive configurations respect preceding instructions, and the last configuration points to the halt instruction.

The computations of M are encoded using timed words over the alphabet $\Sigma_M = \{b_1, b_2, ..., b_n, a_1, a_2\}$. We encode a configuration $\langle i, c, d\rangle$ as a sequence $b_i a_1^c a_2^d$ and a computation as a concatenation of such sequences. Moreover we associate a unit time interval with each configuration with the j-th configuration being encoded in the interval $[j, j + 1)$. In any such interval, the corresponding b occurs at time j, all the a_1's occur in the interval $[j + 0.25, j + 0.5)$ with the first a_1 (if any) occurring at $j+0.25$ and all the a_2's occur in the interval $[j+0.5, j+1)$ with the first a_2 (if any) occurring at $j + 0.5$. We further require that the corresponding a_1's in two such consecutive intervals are a distance one apart. We have a similar requirement for a_2. The second last requirement is a deviation from the reductions in the literature. It is used to ensure that, along with the property L_{ni}, we have no "illegal" insertions of a_1's and a_2's in succeeding configurations.

We now define the timed language L_M which contains exactly the timed words over the alphabet Σ_M which encode a halting computation of M. We define L_M as the intersection of the following timed languages over Σ_M:

1. L_1: words of the form $\vdash b_{i_1} b_{i_2} a_1^* a_2^* b_{i_3} a_1^* a_2^* \cdots b_{i_k} a_1^* a_2^*$ where $i_j \in \{1, \ldots, n\}$, $i_1 = 1$, and $i_k = n$, satisfying: each b_{i_j} occurs at time j; in each interval $[j, j + 1)$, all a_1's occur in the interval $[j + 0.25, j + 0.5)$ with the first a_1 (if any) at $j + 0.25$, and similarly all a_2's occur in the interval $[j + 0.5, j + 1)$ with the first a_2 (if any) at $j + 0.5$
2. L_2: the intersection of the two languages L^{Σ_M,a_1}_{ni} and L^{Σ_M,a_2}_{ni}.
3. L_3: the intersection of the languages L^i_3 for $1 \le i \le n$, where L^i_3 depends on the i-th instruction. For example, if the i-th instruction is: "Increment C and jump to p", then we have that for every b_i at time j, for every a_1 or a_2 at time t in the interval $[j, j+1)$, there is an a_1 or a_2 at time $t + 1$; for every a_1 at time t_1 in $[j, j + 1)$ which is not immediately followed by an a_1, the first symbol in the interval $(t_1 + 1, \infty)$ is an a_1 which is not immediately followed by an a_1 and for every a_2 at time t_2 in $[j, j + 1)$ which is not immediately followed by an a_2, the first symbol in the interval (t_2+1, ∞) is not an a_2; if there is no a_1 in the interval $[j, j + 1)$, then there is an a_1 at time $j + 1.25$ which is not immediately followed by an a_1; if there is no a_2 in the interval $[j, j + 1)$, then there is no a_2 in the interval $[j + 1, j + 2)$; there is a b_p at time $j + 1$.

We now proceed to give the MTL formula φ_M over Σ_M, which describes the language L_M, as the conjunction of the formulas φ_1, φ_2 and φ_3, each describing the languages L_1, L_2 and L_3 respectively. We will use B as a shorthand for $\bigvee_{i\in\{1,\cdots,n\}} b_i$.

1. φ_1 is the conjunction of the formulas:
   $O(b_1 \wedge \neg Oa_1 \wedge \neg Oa_2) \wedge \Box(a_2 \Rightarrow \neg Oa_1) \wedge \Diamond(b_n \wedge \Box_{(0,\infty)}\neg B)$;
   $\Diamond_{[1,1]}B \wedge \neg\Diamond_{(0,1)}\varphi_{act} \wedge \Box((B \wedge \Diamond_{(0,\infty)}B) \Rightarrow (\Diamond_{[1,1]}B \wedge \Box_{(0,1)}\neg B))$;
   and $\Box(B \Rightarrow (\Box_{[0,0.25)}\neg(a_1 \vee a_2) \wedge \Box_{[0.25,0.5)}\neg a_2 \wedge \Box_{[0.5,1)}\neg a_1 \wedge (\Diamond_{(0,1)}a_1 \Rightarrow \Diamond_{[0.25,0.25]}a_1) \wedge (\Diamond_{(0,1)}a_2 \Rightarrow \Diamond_{[0.5,0.5]}a_2)))$.
2. φ_2 is $\varphi_{ni}^{\Sigma_M,a_1} \wedge \varphi_{ni}^{\Sigma_M,a_2}$.
3. $\varphi_3 = \bigwedge_{i\in\{1,\cdots,n\}} \varphi_3^i$, where, for example, for an "increment C and jump to p" instruction i, φ_3^i is the conjunction of the formulas:
   $\Box(b_i \Rightarrow (\Box_{[0,1)}((a_1 \Rightarrow \Diamond_{[1,1]}a_1) \wedge (a_2 \Rightarrow \Diamond_{[1,1]}a_2))))$;
   $\Box(b_i \Rightarrow (\Box_{[0,1)}(((a_1 \wedge \neg Oa_1) \Rightarrow (\Diamond_{[1,1]}O(a_1 \wedge \neg Oa_1))) \wedge ((a_2 \wedge \neg Oa_2) \Rightarrow (\Diamond_{[1,1]}\neg Oa_2)))))$;
   $\Box(b_i \Rightarrow (\Box_{[0,1]}\neg a_1 \Rightarrow \Diamond_{[1.25,1.25]}(a_1 \wedge \neg Oa_1)))$;
   $\Box(b_i \Rightarrow (\Box_{[0,1]}\neg a_2 \Rightarrow (\Box_{[1,2]}\neg a_2)))$;
   and $\Box(b_i \Rightarrow \Diamond_{[1,1]}b_p)$.

It is not difficult to verify that $L^{pw}(\varphi_M)$ coincides with L_M. Now, given a 2-counter machine M, there exists a Turing machine T which outputs the formula φ_M. We note here that T has to "guess" the formula φ_{ni} and apply substitutions based on Σ_M to get the corresponding formula φ_2. This completes the reduction, and we have that the satisfiability problem for MTL^{pw} is undecidable. Since this contradicts the result in [11], there cannot exist an MTL^{pw} formula which describes L_{ni}. This concludes the proof of the strict containment of MTL^{pw} in MTL^{c}.

References

1. R. Alur and D.L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.
2. R. Alur, T. Feder and T.A. Henzinger. The Benefits of Relaxing Punctuality. *Journal of the ACM*, 43(1):116–146, 1996.
3. R. Alur and T.A. Henzinger. Real-time Logics: Complexity and Expressiveness. *Proc. LICS 1990*, 390–401, 1990.
4. P. Bouyer, F. Chevalier and N. Markey. On the expressiveness of TPTL and MTL. *Proc. FSTTCS'05*, volume 3821 of *LNCS*, 432–443, Springer, 2005.
5. D. D'Souza and P. Prabhakar. On the expressiveness of MTL in the pointwise and continuous semantics. Technical Report IISc-CSA-TR-2005-7, IISc, India, 2005
6. D. D'Souza and N. Tabareau. On timed automata with input-determined guards. *FORMATS/FTRTFT*, volume 3253 of *LNCS*, 68–83, Springer, 2004.
7. T.A. Henzinger, J.F. Raskin and P.Y. Schobbens. The Regular Real-Time Languages. *ICALP*, volume 1443 of *LNCS*, 580–591, Springer, 1998.
8. R. Koymans. Specifying Real-Time Properties with Metric Temporal Logic. *Real-Time Systems*, 2(4):255–299, 1990.
9. S. Lasota and I. Walukiewicz. Alternating Timed Automata. *FoSSaCS*, volume 3441 of *LNCS*, 250–265, Springer, 2005.
10. É. Lozes. Adjuncts elimination in the static ambient logic. *Electronic Notes in Theoretical Computer Science*, 96:51–72, 2004.
11. J. Ouaknine and J. Worrell. On the Decidability of Metric Temporal Logic. *Proc. LICS 2005*, 188–197, 2005.