On the Decidability and Complexity of Some Fragments of Metric Temporal Logic

Khushraj Madnani\(^1\), Shankara Narayanan Krishna\(^1\) and Paritosh K. Pandya\(^2\)

\(^1\)IIT Bombay, Powai, Mumbai, India 400 076
\(^2\)Tata Institute of Fundamental Research, Colaba, Mumbai, India 400

\{khushraj,krishnas\}@cse.iitb.ac.in, pandya@tcs.tifr.res.in

Abstract. Metric Temporal Logic, MTL\(_{pw}[^U I, S I]\), is amongst the most studied real-time logics. It exhibits considerable diversity in expressiveness and decidability properties based on the permitted set of modalities and the nature of time interval constraints \(I\). In this paper, we sharpen the decidability results by showing that the satisfiability of MTL\(_{pw}[^U I, S_{NS}]\) (where \(NS\) denotes non-singular intervals) is also decidable over finite pointwise strictly monotonic time. We give a satisfiability preserving reduction from MTL\(_{pw}[^U I, S_{NS}]\) to the decidable logic MTL\(_{pw}[^U I]\) of Ouaknine and Worrell using a novel technique of temporal projections with oversampling. We also investigate the decidability of unary fragment MTL\(_{pw}[^\Diamond I, \Diamond S]\) and we compare the expressive powers of some of these fragments.

1 Introduction

Real time logics specify properties of how system state evolves with time and where quantitative time distance between events is significant. Metric Temporal Logic (MTL) introduced by Koymans \[8\] is a prominent real-time logic. In this logic, the temporal modalities \(U_I\) and \(S_I\) are constrained by a time interval \(I\). MTL exhibits considerable diversity in expressiveness and decidability based on the permitted set of modalities and the nature of time interval constraints \(I\).

The classical results of Alur and Henzinger showed that satisfiability of MTL\(\{U_I, S_I\}\) as well as its model checking problem against timed automata are undecidable in general \[1, 5\]. In their seminal paper \[2\], the authors proposed a sublogic MTL\(\{U_{NS}, S_{NS}\}\) having only non-singular intervals \(NS\), where the satisfiability is decidable with \textit{EXPSPACE} − \textit{complete} complexity. The satisfiability of MTL\(\{U_I\}\) was considered to be undecidable for a long time, until Ouaknine and Worrell \[11\] proved that the satisfiability of MTL\(_{pw}\{U_I\}\) over finite timed words is decidable, albeit with a non-primitive recursive lower bound. Subsequently, in \[12\], it was shown that over infinite timed words, the satisfiability of MTL\(_{pw}\{U_I\}\) is undecidable. The satisfiability of MTL\(\{U_I\}\) over continuous time is also undecidable.

In this paper, we sharpen the decidability results for fragments of MTL\(_{pw}\{U_I, S_I\}\). We consider the logic MTL\(_{pw}\{U_I, S_{NS}\}\) which has been shown \[13\] to be strictly more expressive than the known decidable fragments MTL\(\{U_{NS}, S_{NS}\}\) as well
as MTL[SI], but strictly less expressive than the full MTL[U, SI]. As our main result, we show that satisfiability of MTLpw[U, SNS] is decidable over pointwise strictly monotonic time (i.e. finite timed words). By symmetry it is easy to show that MTLpw[U, SI] also has decidable satisfiability. The result is established by giving a satisfiability preserving reduction from the logic MTLpw[U, SNS] to MTLpw[U] of Ouaknine and Worrell using a novel technique of temporal projections with oversampling.

Temporal projection is a technique which allows obtaining equi-satisfiable formula in a more restricted logic by using additional auxiliary propositions. The formula transformation is carried out in a conservative fashion so that the models of the original formula and those of the transformed formula are related in projection-embedding fashion. This technique was used in a number of works on continuous time temporal logics to obtain equi-satisfiable formulae with only restricted set of modalities (e.g. from MTL[U, SI] to MTL[U]). [6,3,15,7]. In this paper, we generalize the technique to pointwise models where transformed formula is interpreted over timed words which are “oversampling” of the original timed words. Thus, model embedding involves adding intermediate “non-action” time points in the timed word where only the auxiliary propositions are interpreted.

Our transformation of the MTLpw[U, SNS] formula to MTLpw[U] formula relies upon the properties of the unary modality ♦NS. In a recent work [14], Pandya and Shah formulated a “horizontal stacking” properties of the bounded ♦[l,u]φ modality: this allows the truth of ♦[l,u]φ at a point in any unit interval to be related to the first and the last occurrences of φ in some related unit intervals. Our transformation builds upon these properties to achieve elimination of past modalities using temporal projection with oversampling.

Several real-time properties can be specified using only the unary future ♦I, and past ♦I operators. As our second main contribution, we investigate the decidability of unary fragment MTL[♦I, ♦I] (this question was posed by A. Rabinovich in a personal communication). We show that MTLpw[♦I, ♦I] over finite pointwise time is undecidable, whereas MTLpw[♦I] over finite pointwise models already has Ackermann-hard satisfiability checking. Hence, restriction to unary modalities does not improve the decidability properties of MTLpw[U, SI]. We compare the expressive powers of some of these fragments using the technique of EF games for MTL [13].

2 Metric Temporal Logic

In this section, we describe the syntax and semantics of MTL in the pointwise sense. The definitions below are standard. Let Σ be a finite alphabet of events. A finite timed word over Σ is a sequence ρ = (A1, t1)(A2, t2)...(An, tn) where A₁ ⊆ Σ, A₁ ∩ Σ ≠ ∅, and ti ∈ R≥0 for 1 ≤ i ≤ n. Further, ti < tj for all 1 ≤ i < j ≤ n. BP and EP are special endmarkers which hold at the start and end point of any timed word. We use the short form ρ(σ, τ) to represent a timed word; σ = A1A2...An and τ = t1t2...tn. Each point ρ(i), 1 ≤ i ≤ n is
called an action point, and let \( \text{dom}(\rho) = \{1, 2, \ldots, n\} \) be the set of positions of \( \rho \). Let \( \text{time}(\rho) \) denote the time stamp of the last action point of \( \rho \). Let \( \sigma_i = A_i \) and \( \tau_i = t_i \). Given \( \Sigma \), the formulae of MTL are built from \( \Sigma \), modalities \( BP, EP \) using boolean connectives and time constrained versions of the modalities \( U \) and \( S \) as follows:

\[
\varphi := a(\in \Sigma) | BP | EP | \text{true} | \varphi \land \varphi | \neg \varphi | \varphi \cup_1 \varphi | \varphi \cup_t \varphi
\]

where \( I \) is an open, half-open or closed interval with end points in \( \mathbb{N} \cup \{\infty\} \).

Point-Wise Semantics: Given a finite timed word \( \rho \), and an MTL formula \( \varphi \), in the pointwise semantics, the temporal connectives of \( \varphi \) quantify over a countable set of positions in \( \rho \). For an alphabet \( \Sigma \), a timed word \( \rho = (\sigma, \tau) \), a position \( i \in \text{dom}(\rho) \), and an MTL formula \( \varphi \), the satisfaction of \( \varphi \) at a position \( i \) of \( \rho \) is denoted \((\rho, i) \models \varphi\), and is defined as follows:

\[
\begin{align*}
(\rho, i) &\models a & \iff & a \in \sigma_i \\
(\rho, i) &\models \neg \varphi & \iff & (\rho, i) \not\models \varphi \\
(\rho, i) &\models \varphi_1 \land \varphi_2 & \iff & (\rho, i) \models \varphi_1 \land (\rho, i) \models \varphi_2 \\
(\rho, i) &\models \varphi_1 \cup_1 \varphi_2 & \iff & \exists j > i, \rho, j \models (\varphi_2, t_j - t_i \in I, \text{and } \rho, k \models \varphi_1 \forall i < k < j) \\
(\rho, i) &\models \varphi_1 \cup_t \varphi_2 & \iff & \exists j > i, \rho, j \models (\varphi_2, t_i - t_j \in I, \text{and } \rho, k \models \varphi_1 \forall j < k < i)
\end{align*}
\]

\( \rho \) satisfies \( \varphi \) denoted \( (\rho, 0) \models \varphi \text{ iff } (\rho, 0) \models \varphi \). Let \( L(\varphi) = \{ \rho | (\rho, 0) \models \varphi \} \). The set of all timed words over \( \Sigma \) is denoted \( T\Sigma^* \). Additional temporal connectives are defined in the standard way: we have the constrained future and past eventuality modalities \( \boxdot \) and \( \blacklozenge \), interpreted in the pointwise semantics having modalities \( \varrho, \iota \) denoted \((\rho, i)\).

\[\begin{align*}
\varrho &\equiv \text{true} U_a \land \varphi_1 \land \varphi_2 & \equiv & \varphi_1 \cup_1 \varphi_2 \\
\blacklozenge &\equiv \varphi_1 \cup_t \varphi_2 & \equiv & \varphi_1 \cup_1 \varphi_2 \\
\boxdot &\equiv \varphi_1 \cup_1 \varphi_2 & \equiv & \varphi_1 \cup_t \varphi_2 \\
\blacklozenge &\equiv \varphi_1 \cup_t \varphi_2 & \equiv & \varphi_1 \cup_1 \varphi_2
\end{align*}\]

\( \varrho \text{ and } \blacklozenge \) denote the satisfaction of \( \varphi \) at a position \( i \) of \( \rho \). 

3 Decidability of MTL\textsuperscript{pw}[U\textsubscript{1}, S\textsubscript{NS}]

In this section, we show that the class MTL\textsuperscript{pw}[U\textsubscript{1}, S\textsubscript{NS}] is decidable, by giving a satisfiability preserving reduction to MTL\textsuperscript{pw}[U\textsubscript{1}]. Two important transformations needed in this reduction are flattening of the formula (which removes nesting of temporal operators using auxiliary propositions) and oversampling closure which makes the truth of the formula invariant under insertion of additional time points (oversampling).

3.1 Temporal Projections

Let \( \Sigma' \supseteq \Sigma \). Consider a word \( \rho' = (Y_1, 0)(Y_2, t_2') \ldots (Y_m, t_m') \in T\Sigma'^* \). Then, \( \rho' \upharpoonright \Sigma \) is the timed word \( \rho \in T\Sigma^* \) obtained by the steps (E1) followed by (E2):

(E1) Erase all symbols of \( \Sigma' \setminus \Sigma \) from \( \rho' \). Call the resultant word \( \rho'' \).

(E2) Erase all symbols of the form \( (\emptyset, t_i) \) from \( \rho'' \), to obtain \( \rho \). Given \( \rho' \) as above, and a position \( i \) of \( \rho' \), \( i \) is called an action-point iff \( Y_i \cap \Sigma \neq \emptyset \). For \( \rho = (X_1, 0)(X_2, t_2) \ldots (X_n, t_n) \) a timed word in \( T\Sigma^* \), \( \rho' \) is called a \( \Sigma' \)-oversampling of \( \rho \), denoted by \( \rho' \downarrow \Sigma' = \rho \) iff (i) \( \rho' \upharpoonright \Sigma = \rho \), and (ii) \( Y_1 \cap \Sigma \neq \emptyset \) and \( Y_m \cap \Sigma \neq \emptyset \).
In this case \( \rho \) is called a projection of \( \rho' \). A \( \Sigma' \)-oversampling \( \rho' \) is called a **simple extension** of \( \rho \), denoted by \( \rho' \upharpoonright_{\Sigma'} = \rho \) if \( m = n \). For a \( \Sigma' \)-oversampling \( \rho' \) of \( \rho \), we can map the action points to the points of \( \rho \) as follows: \( \rho'(1) = 1 \). For \( 1 < i < m \), \( \rho'(i) = j \) iff (i) \( Y_i \cap \Sigma = X_j \), \( t'_i = t_j \), and (ii) For some \( h > 0 \), \( \rho'(i-h) = j-1 \) iff \( Y_i \cap \Sigma = \emptyset \), for all \( r, i-h < r < i \), and \( Y_{i-h} \cap \Sigma = X_{j-1} \). For simple extension, this mapping is the identity.

The above notions can be extended to timed languages. For \( L \subseteq T\Sigma^* \) and \( L' \subseteq T\Sigma'^* \), \( L' \upharpoonright_{\Sigma'} = L \) iff for every \( \rho' \in L' \), we have some \( \rho \in L \) such that \( \rho' \upharpoonright_{\Sigma'} = \rho \), and for every \( \rho \in L \), there exists some \( \rho' \in L' \) such that \( \rho' \upharpoonright_{\Sigma'} = \rho \).

**Example 1.** Let \( \Sigma = \{a, b\} \) and let \( \Sigma' = \{a, b, c\} \). A \( \Sigma' \)-oversampling of \( \rho = (\{a\}, 0)(\{a, b\}, 0.3)(\{a\}, 4.8) \) is \( \rho_1 = (\{a, c\}, 0)(\{c\}, 0.1)(\{a, b\}, 0.3)(\{a, c\}, 4.8) \). \( \rho_2 = (\{a, c\}, 0)(\{b, c\}, 0.1)(\{a, b\}, 0.3)(\{a, c\}, 4.8) \) is not a \( \Sigma' \)-oversampling of \( \rho \). \( \rho_1 \upharpoonright \{a, b\} = \rho \), while \( \rho_2 \upharpoonright \{a, b\} \neq \rho \). Neither \( \rho_1 \) nor \( \rho_2 \) is simple extension of \( \rho \). Also, \( \rho_1(1) = 1, \rho_1(3) = 2, \rho_1(4) = 3 \).

**Flattening** Let \( \varphi \in \text{MTL}^{\text{pw}}[U_1, S_I] \) over \( \Sigma \). Given any sub-formula \( \psi_i \) of \( \varphi \), and a fresh symbol \( a_i \), the equivalence \( X_i = \overline{\psi_i} \leftrightarrow a_i \) is called a temporal definition and \( a_i \) is called a witness. Let \( \psi = \varphi[a_i/\psi_i] \) be the formula obtained by replacing all occurrences of \( \psi_i \) in \( \varphi \), with the witness \( a_i \). Flattening is done recursively until we have replaced all future/past modalities of interest with witness variables, obtaining \( \varphi_{\text{flat}} = \psi \land X \), where \( X \) is the conjunction of all temporal definitions. Let \( \Sigma' \) be the set of auxiliary witness propositions used in \( X \), and let \( \Delta = \Sigma \cup \Sigma' \). Let \( \text{act} = \bigvee_{a \in \Sigma} a \).

**Lemma 1.** Let \( \varphi \in \text{MTL}^{\text{pw}}[U_1, S_I] \) over \( \Sigma \) and let \( \varphi_{\text{flat}} \) be the flattened formula obtained from \( \varphi \). Note that \( \varphi_{\text{flat}} \) is over \( \Delta \). Let \( \rho \) be a timed word in \( T\Sigma^* \). For any \( \Delta \)-extension \( \rho' \) of \( \rho \) such that \( \rho' \upharpoonright_{\Delta} = \rho \), we have \( \rho \models \varphi \iff \rho' \models \varphi_{\text{flat}} \).

Note that \( \varphi \) and \( \varphi_{\text{flat}} \) may not be equisatisfiable unlike in MTL over the continuous semantics.

**Lemma 2 (oversampling closure).** Let \( \varphi \in \text{MTL}^{\text{pw}}[U_1, S_I] \) over \( \Sigma \) and \( \Sigma' \supseteq \Sigma \). Then \( L(\varphi) = L(\varphi) \upharpoonright_{\Sigma'} \), where \( \varphi = \varphi' \land \((\text{BP} \Rightarrow \text{act}) \land (\text{EP} \Rightarrow \text{act})) \), and \( \varphi' \) is obtained from \( \varphi \) by replacing recursively all subformulae of the form \( (a_i U_t a_j) \) with \( (\text{act} \Rightarrow a_i) U_t (a_j \land \text{act}) \) \( (\text{act} \Rightarrow a_i) \text{S}_t (a_j \land \text{act}) \).

**Proof.** In Appendix [A]

**Lemma 3.** Let \( \varphi \in \text{MTL}^{\text{pw}}[U_1, S_I] \) over \( \Sigma \), and \( \Sigma' \supseteq \Sigma \). Then \( L(\varphi) = L(\varphi_{\text{flat}}) \upharpoonright_{\Sigma'} \), where \( \varphi_{\text{flat}} \) is obtained from \( \varphi_{\text{flat}} \) using Lemma [2].

Given a formula \( \varphi \) (over \( \Sigma \)) of logic \( L_1 \), we can often find a formula \( \psi \) (over \( \Sigma' \)) of a much simpler/desirable logic \( L_2 \) such that \( L(\varphi) = L(\psi) \upharpoonright_{\Sigma'} \). We say that \( \varphi \) is equivalent modulo temporal projections (equisatisfiable) to \( \psi \). This is denoted \( \psi \upharpoonright \Sigma \equiv \varphi \). Example [2] illustrates this.
We now give a satisfiability preserving reduction from $\mathbf{MTL}^{pw}[\mathcal{U}_t, S_{NS}]$ to logic $\mathbf{MTL}^{pw}[\mathcal{U}_t]$. Fix a formula $\phi \in \mathbf{MTL}^{pw}[\mathcal{U}_t, S_{NS}]$ over propositions $\Sigma$. For simplicity we assume that $S_{NS}$ only has interval $NS$ which are left-closed-right-open, e.g. $[3, \infty)$ or $(5, 17)$. Other forms of intervals can be handled similarly to the reduction given below.

- Given $\phi \in \mathbf{MTL}^{pw}[\mathcal{U}_t, S_{NS}]$ we transform the formula to an equivalent formula $\phi' \in \mathbf{MTL}^{pw}[\mathcal{U}_t, \hat{\mathcal{U}}_{\nu}]$ where $\nu$ is either infinite (i.e. $[l, \infty)$) or unit (i.e. $[l, l+1)$). Standard techniques (see [15, 17]) apply to give this reduction which can also be found in Theorem 4

- Removal of $\hat{\mathcal{U}}_{|l,l+1]}$ modality requires us to consider behaviours where additional non-action time points have to be added. Each occurrence of the operator gives its own requirement of adding time points. Hence, we consider equi-satisfiable $\hat{\phi}_{flat}$ which is invariant under such oversampling.

- Let $\phi_{flat}$ over $\Sigma'$ be obtained by flattening and oversampling closure as in Lemma 3. This formula consists of a conjunction of temporal definitions. Lemma 4 below shows how temporal definition with past operator of the form $\Box[act \Rightarrow (b \Leftrightarrow \hat{\phi}_{[l,\infty)}(a \land act))]$ can be replaced by an equivalent formula in $\mathbf{MTL}^{pw}[\mathcal{U}_t]$.

- Similarly, Lemma 5 gives elimination of temporal definition involving $\hat{\mathcal{U}}_{|l,l+1]}$ operator using an equi-satisfiable $\mathbf{MTL}^{pw}[\mathcal{U}_t]$ formula.

- The above constitute the main lemmas of our proof. By repeatedly applying them we get an equi-satisfiable formula of $\mathbf{MTL}^{pw}[\mathcal{U}_t]$.  

**Lemma 4.** Consider a temporal definition $\hat{X}_{[l,\infty)} = \Box[act \Rightarrow (b \Leftrightarrow \hat{\phi}_{[l,\infty)}(a \land act))]$. Then we can synthesize a formula $\nu \in \mathbf{MTL}^{pw}[\mathcal{U}_t]$ equivalent to $\hat{X}_{[l,\infty)}$.

**Proof.** Let $\alpha = (act \Rightarrow (-a \land -b))$. Consider the following formulae in $\mathbf{MTL}^{pw}[\mathcal{U}_t]$:

1. $\varphi_1 : [\Box a \lor \{a \hat{\mathcal{U}}[(a \land act) \land \Box_{[l,1]}(act \Rightarrow -b))]\}
2. \varphi_2 : \Box[(a \land act) \Rightarrow \Box_{[l,\infty)}(act \Rightarrow b)]$.

Let $\nu = \varphi_1 \land \varphi_2$. We claim that $\rho' \models \hat{X}_{[l,\infty)}$ iff $\rho' \models \nu$ for any $\rho' \in T\Sigma^{eq}$.

Assume $\rho' \models \hat{X}_{[l,\infty)}$. Assume to contrary that $\rho' \models \neg \varphi_1$. Then, either there is a point act marked $b$ before the first occurrence of $a \land act$, or there is a point
that we have all the integral time points, we have a proposition required to hold at some possibly non-action points. Further, to make sure marking points with specifies the behaviour of \( c \) (using only future modalities, we need auxiliary propositions \( c, \text{beg}, \text{end} \) by the following formula:

\[
\text{Proof.} \quad \text{Firstly, notice that if there exists } \psi, \text{ Lemma 6. Consider a temporal definition } X_{[l,l+1]} = \Box[\text{act } \Rightarrow (b \Leftrightarrow \Box_{[l,l+1]}(a \land \text{act}))) \text{ whose defining modality is } \Box_{[l,l+1]}. \text{ In Lemma 6 we show how to synthesize a formula } \psi \in \text{MTL}^{\text{pw}}[\mathcal{U}_l] \text{ which is equisatisfiable to } X_{[l,l+1]}. \text{ For this, we construct an oversampling } \rho' \text{ of } \rho \text{ over an extended alphabet } \Sigma''.

\text{Lemma 5 (14). Given a timed word } \rho = (\sigma, \tau), \text{ integers } l, t, \text{ and an point } i \in \text{dom}(\rho). \text{ For } \tau_i \in [t + l + 1, t + l + 2), \text{ we have } \rho, i \models \Box_{[l,l+1]}(a \land \text{act}) \text{ iff }

- \tau_i > \mathcal{F}^{a}_{[l+1,l+1]}(\rho) \land \tau_i \in [t + l + 1, \mathcal{L}^{a}_{[l+1,l+1]}(\rho) + l + 1), \text{ or }
- \tau_i > \mathcal{F}^{a}_{[t+1,l+2]}(\rho) \land \tau_i \in (\mathcal{F}^{a}_{[t+1,l+2]}(\rho) + l + 1, t + l + 2).

\text{Consider a temporal definition, } X_{[l,l+1]} = \Box[\text{act } \Rightarrow (b \Leftrightarrow \Box_{[l,l+1]}(a \land \text{act}))) \text{ whose defining modality is } \Box_{[l,l+1]}. \text{ In Lemma 6 we show how to synthesize a formula } \psi \in \text{MTL}^{\text{pw}}[\mathcal{U}_l] \text{ which is equisatisfiable to } X_{[l,l+1]}. \text{ For this, we construct an oversampling } \rho' \text{ of } \rho \text{ over an extended alphabet } \Sigma''.

\text{Lemma 6. Consider a temporal definition } X_{[l,l+1]} = \Box[\text{act } \Rightarrow (b \Leftrightarrow \Box_{[l,l+1]}(a \land \text{act}))) \text{ whose defining modality is } \Box_{[l,l+1]}. \text{ In Lemma 6 we show how to synthesize a formula } \psi \in \text{MTL}^{\text{pw}}[\mathcal{U}_l] \text{ which is equisatisfiable to } X_{[l,l+1]}. \text{ For this, we construct an oversampling } \rho' \text{ of } \rho \text{ over an extended alphabet } \Sigma''.

1. \rho' \models \psi \Rightarrow \rho' \models X_{[l,l+1]}
2. \rho' \models X_{[l,l+1]} \Rightarrow \exists \rho'' \in T\Sigma'' \text{ such that } \rho'' \models \psi, \text{ and } \rho'' \upharpoonright_{\Sigma''} = \rho' \upharpoonright_{\Sigma''}.

\text{Proof. Firstly, notice that if there exists } i \text{ in } \rho \text{ marked } \text{act } \land a, \text{ then all points } j \text{ marked } \text{act} \text{ such that } t_j \in [t_i + l, t_i + l + 1) \text{ must be marked } b. \text{ This is enforced by the following formula:}

- \varphi_8 : \Box[(a \land \text{act}) \Rightarrow \Box_{[l,l+1]}(\text{act } \Rightarrow b)]

Marking points with } \neg b \text{ is considerably more involved. At a time point } t_i \in [t + l + 1, t + l + 2), \text{ holds only if there is an } a \land \text{act} \text{ in the interval } (t_i - l - 1, t_i - l) \subseteq [t, t + 2). \text{ Here we exploit Lemma 5. But to state its conditions using only future modalities, we need auxiliary propositions } c, \text{beg}_b, \text{end}_b \text{ which are required to hold at some possibly non-action points. Further, to make sure that we have all the integral time points, we have a proposition } c \text{ that marks every integer valued time point within the time span of } \rho. \text{ The following formula specifies the behaviour of } c. \text{ Note that } c \text{ is uniquely determined by the formula:}

- \varphi_1 : c \land \Box[\Box c \land \neg \text{EP} \Rightarrow \Box_{(0,1)} \neg c \land [\Diamond_{(1,1)} c \lor \Diamond_{(0,1)} \text{EP}]]
To see the need for $beg_b$, $end_b$, for some $t \in \mathbb{N}$, consider the case where the last $act \land a$ in $(t, t + 1]$ occurs at $u$ and the first $act \land a$ in $[t + 1, t + 2)$ occurs at $v$. If $v - u > 1$, then all points $act$ in $[u + l + 1, v + l)$ must be marked $\neg b$. See Figure 4. To facilitate this marking correctly, we introduce a non-action point marked $end_b$ at $v + l$, and a non-action point marked $beg_b$ at $u + l + 1$ in $\rho'$. The following formulae assert that $end_b$ holds at distance $t$ from the first $a$ in each unit interval with integral end points. The first such $end_b$ happens beyond $(0, l)$:

$\varphi_2 : \Box[(c \land \delta_{[0,1]}(a \land act)) \Rightarrow [(act \Rightarrow \neg a) \overline{U}_{[0,1]}((a \land act) \land [\delta_{[0,1]}end_b \lor \delta_{[0,1]}EP)])]$  

$\varphi_3 : \Box_{[0,1]}\neg end_b$ (if $l \neq 0$)

In a similar way, the following formulae assert that $beg_b$ holds at distance $l + 1$ from the last $a$ in each unit interval with integral end points. The first such $beg_b$ happens beyond $[0, l + 1)$:

$\varphi_4 : \Box[(c \land \delta_{[0,1]}(a \land act)) \Rightarrow \delta_{[0,1]}\{(a \land act) \land [(act \Rightarrow \neg a) \land \neg c]Uc\land (\delta_{[t + 1,l + 1]}beg_b \lor \delta_{[0,l + 1]}EP)\}$  

$\varphi_5 : \Box_{[0,l + 1]}\neg beg_b$

The following formulae assert that each unit interval with integral end points can have at most one $end_b$, and one $beg_b$ : if a unit interval $[t, t+1)$, with $t \in \mathbb{N}$ has no $a$, then there is no $end_b$ in the interval $[t+l, t+l+1)$, and there is no $beg_b$ in the interval $[t+l+1, t+l+2)$.

$\varphi_6 : c \land \Box_{[0,1]}(act \Rightarrow \neg a) \Rightarrow \Box_{[0,1]}\neg end_b \land \Box_{[t+l+1]}\neg beg_b$

$\varphi_7 : c \land \Box_{[0,1]}x \Rightarrow (x \neg \overline{U}_{[0,1]}[x \land (\neg x \land \neg c)U_{[0,1]}(c \lor EP)])$ for $x \in \{end_b, beg_b\}$.

Note that above formulae uniquely determine the points where $c, end_b, beg_b$ must hold in $\rho'$ based on where $a$ holds in $\rho'$. Using these extra propositions, we now construct a formula which enforce the other direction $act \Rightarrow (b \Rightarrow \delta_{[l,l+1]}(a \land act))$ of $X_{[l,l+1]}$ within interval $[t+l+1, t+l+2)$. We sketch this proof case-wise.

**Case 1**: If $act \land \neg a$ holds throughout $(t, t+2)$, then $\delta_{[l,l+1]}(a \land act)$ cannot hold anywhere in $[t+l+1, t+l+2)$ (if it did, then we will have an $a \land act$ in $(t, t+2)$).

**Case 2**: If $\neg a$ holds at all points $act$ in $(t, t+1)$, and there is an $act \land a$ in $(t+1, t+2)$. Assume that the first $a \land act$ in $(t+1, t+2)$ occurs at $s = t+1 + \epsilon$. Then, by $\varphi_2$, we have a $end_b$ at $s+l = t+1 + \epsilon + l$. Also, $\neg beg_b$ holds throughout $(t+l+1, t+l+2)$. $\delta_{[l,l+1]}(a \land act)$ cannot hold at points $act$ in $(t+l+1, s+l)$, for it did, then we must have an $a \land act$ in $(t, s)$. The formula $\varphi_9$ considers cases 1 and 2.

$\varphi_9 : \Box[(c \land \Box_{[0,1]}\neg beg_b) \Rightarrow ((act \Rightarrow \neg b) \land \neg x)\overline{U}x]$ where $x = (end_b \lor c \lor EP)$.

**Case 3** If $\neg a$ holds at all points $act$ in $[t+1, t+2)$, and if there is an $act \land a$ in $(t, t+1)$. Assume that the last $a \land act$ in $(t, t+1)$ occurred at $t+\delta = v, 0 \leq \delta < 1$. 

To determine the points where $\neg a$ and $\neg c$ hold in $[t+1, t+2)$. Since $a \land act$ is not possible in $(t+1, t+2)$, the only possibility is $\neg a$ and $\neg c$ hold in $[t+1, t+2)$. The formula $\varphi_{10}$ considers cases 1 and 2.

$\varphi_{10} : \Box[(c \land \Box_{[0,1]}\neg beg_b) \Rightarrow ((act \Rightarrow \neg b) \land \neg x)\overline{U}x]$ where $x = (end_b \lor c \lor EP)$.
Then by $\varphi_{10}$, we have $b$ holds at all points $act$ in $[v + l, v + l + 1]$. Also, by $\varphi_4$, $beg_b$ holds at $v + l + 1$, and $\neg end_b$ holds throughout $[t + l + 1, t + l + 2]$ by $\varphi_6$. However, we cannot have a $b \land act$ in $[v + l + 1, t + l + 2)$, for this would mean the presence of an $a \land act$ in $(v, t + 2)$. Note that if the last $a \land act$ of $[t, t + 1)$ is at $t$, then $beg_b$ holds at $t + l + 1$.

Case 4 If there is an $a \land act$ in both $[t + 1, t + 2)$ and $[t, t + 1)$. Assume that the last $a \land act$ in $[t, t + 1)$ is at $u = t + \epsilon$, and the first $a \land act$ in $[t + 1, t + 2)$ is at $v = t + 1 + \kappa$, with $\epsilon, \kappa \geq 0$. If $v - u \leq 1$, then $\epsilon \geq \kappa$, and by $\varphi_8$, we have $b$ holds at all points $act$ in $[t + \epsilon + l, t + l + 2 + \kappa)$. However, if $v - u > 1$, then $\kappa > \epsilon$, and by $\varphi_8$, $b$ holds at all points $act$ of $[u + l, u + l + 1)$ and $[v + l, v + l + 1)$, with $u + l + 1 < v + l$. In this case, all points $act$ in the range $[u + l + 1, v + l)$ must be marked $\neg b$. The following formula handles cases 3 and 4. For $x = \neg (end_b \lor c \lor EP)$,
− \varphi_{10} : \Box \{ (c \land \neg end_b \U [0,1] beg_b) \} \Rightarrow \Box [0,1] (beg_b \land (((act \Rightarrow \neg b) \land x) \U (end_b \lor c \lor EP))\} \}

Let \psi = \bigwedge_{i=1}^{10} \varphi_i \in MTL^\omega [\U] .

**Proof of 1.** We claim that \rho', i \models \psi implies \rho', i \models \mathcal{X}_{l,l+1} . Assume that \rho', i \models \psi, and consider a point \(i\). Let \(t_i \in [t+l, t+l+1]\) for some \(t \in \mathbb{N}\). Suppose \(\rho', i \not\models \mathcal{X}_{l,l+1}(a \land act)\) and \(\rho', i \models act\). We show that \(\rho', i \not\models \neg b\).

Since \(\rho', i \not\models \mathcal{X}_{l,l+1}(a \land act)\), all points marked act in \((t_i - l - 1, t_i - l)\) are marked \(\neg a\). Note that \((t_i - l - 1, t_i - l) \subset [t - 1, t + 1)\) with \(t - 1 \leq t_i - l - 1 < t\), and \(t \leq t_i - l < t + 1\).

1. We have \(\Box \neg a\) in \((t_i - l - 1, t_i - l)\). Assume that there is an \(a \land act\) in \([t - 1, t)\), and the last such occurs at \(u \leq t_i - l - 1\). Assume further that there is an \(a\) in \([t, t + 1)\), and the first such \(a \land act\) occurs at \(v = t_i - l\). Then, by case 4 of the analysis, we obtain \(\Box \neg b\) at all points act in \([u + l + 1, v + l)\). Clearly, \(u + l + 1 \leq v < v + l\), hence \(act \land \neg b\) holds at \(t_i\).

2. Assume that there is no \(a \land act\) in \([t - 1, t)\), but there is an \(a \land act\) in \([t, t + 1)\). The first such \(a \land act\) occurs at \(s > t_i - l\). Then, by case 2 of our analysis, \(\neg b\) holds at all points act in \([t + l, s + l)\). Clearly, \(t + l \leq t_i < s + l\), hence, \(act \land \neg b\) holds at \(t_i\).

3. Assume that there is an \(a \land act\) in \([t - 1, t)\), and the last such occurs at \(v \leq t_i - l - 1\). Further, assume there is no \(a \land act\) in \([t, t + 1)\). Then, by case 3 of our analysis, \(\neg b\) holds at all points act of \([v + l + 1, t + l + 1)\). Clearly, \(v + l + 1 \leq t_i < t + l + 1\), hence \(act \land \neg b\) holds at \(t_i\).

4. Assume that there is no \(a \land act\) in both \([t - 1, t)\) as well as \([t, t + 1)\). In this case, by case 1, \(\neg b\) holds at all points of \([t + l, t + l + 1)\). Clearly, \(t_i \in [t + l, t + l + 1)\), hence \(act \land \neg b\) holds at \(t_i\).

Thus, \(\rho', i \models \neg b\), and hence \(\rho', i \models (act \Rightarrow (\neg \mathcal{X}_{l,l+1}(a \land act) \Rightarrow b))\).

Now assume that \(\rho', i \models act \land \neg b\). We show that \(\rho', i \models \neg \mathcal{X}_{l,l+1}(a \land act)\). Suppose \(\rho', i \models \mathcal{X}_{l,l+1}(a \land act)\). There is a point \(t \in (t_i - l - 1, t_i - l)\) where \(a \land act\) holds. Then, by \(\varphi_8\), we have \((act \Rightarrow b)\) holds at all points of \([t + l, t + l + 1)\). Note that \(t_i \in [t + l, t + l + 1)\), and henceforth \(\rho', i \models (act \Rightarrow b)\), which contradicts the assumption we started out with. Hence, \(\rho', i \models \neg \mathcal{X}_{l,l+1}(a \land act)\).

Now consider the case of a point act at \(t_i \in [0, l)\). Clearly, for such a \(t_i\), \(\mathcal{X}_{l,l+1}(a \land act)\) cannot hold. \(\varphi_2 - \varphi_7\) assert that (i) there is no \(end_b\) in \([0, l)\) and there is no \(beg_b\) in \([0, l + 1)\), (ii) if in some unit interval with integral end points, there is no \(end_b\) and \(beg_b\), then in that interval all points act will be marked \(\neg b\). Thus, in \([0, l)\) all points act are marked \(\neg b\). At timestamps \(t \geq l\), all points act satisfying \(\mathcal{X}_{l,l+1}(a \land act)\) are marked \(b\) by \(\varphi_8\). We have thus showed \(\rho', i \models \psi\) implies \(\rho', i \models \mathcal{X}_{l,l+1}\).

**Proof of 2.** Assume that \(\rho' \models \mathcal{X}_{l,l+1}\). Then we can construct \(\rho'' \in T \Sigma^\omega\) such that \(\rho'' \models \psi\), and \(\rho'' \downarrow_{\Sigma^\omega} = \rho' \downarrow_{\Sigma^\omega}\). Assume that for any \(\rho'' \in T \Sigma^\omega\), \(\rho' \models \mathcal{X}_{l,l+1}\). Then, at any point \(i\) of \(\rho'\), \(\rho', i \models act\) iff \(\rho', i \models (b \Rightarrow (\mathcal{X}_{l,l+1}(a \land act)))\).

Consider the word \(\hat{\rho} = \rho' \downarrow_{\Sigma^\omega}\). \(\rho' \models \mathcal{X}_{l,l+1}\) implies \(\hat{\rho} \models \mathcal{X}_{l,l+1}\). From \(\hat{\rho}\), construct as given by the formulæ \(\varphi_1\) to \(\varphi_7\) of Lemma 6, the oversampling
\[ \rho'' \in T\Sigma'' \]. That is, \( \rho'' \models \bigwedge_{i=1}^n \varphi_i \land \hat{X}_{[l,l+1]} \). Now, we first show that \( \rho'' \models \psi \). If not, then \( \rho'' \models \neg \varphi_8 \lor \neg \varphi_9 \lor \neg \varphi_{10} \).

(a) Assume \( \rho'' \models \neg \varphi_8 \). Then, there exists \( i \) such that \( \rho'', i \models (a \land \varphi_1) \), and \( \rho'', i \models \bigwedge_{[l,l+1]} (\varphi_2 \land \neg b) \). Let \( t_j \in \{t_i + l, t_i + l + 1\} \) be the point where \( \varphi_2 \land \neg b \) holds. Then, we have \( \rho'', j \models (a \land \neg b) \land \bigwedge_{[l,l+1]} (a \land \varphi_1) \) (recall that \( t_i \in \{t_j - l - 1, t_j - l\} \), and \( \rho'', i \models (a \land \varphi_1) \)). That is, \( \rho'', j \models \varphi_1 \land \varphi_2 \land \neg b \) and \( \rho'', j \models (a \land \varphi_1) \). Hence, \( \rho'' \models \neg \hat{X}_{[l,l+1]} \), a contradiction.

(b) Assume \( \rho'' \models \neg \varphi_9 \). Then, there exists an integral point \( i \) such that \( \rho'', i \models (c \land \square_{[0,1]} \neg \text{beg}_b) \) and \( \rho'', i \models ((a \implies \neg b) \land \neg x) \). By \( \varphi_4, \varphi_6, \rho'', i \models (c \land \square_{[0,1]} \neg \text{beg}_b) \) implies that there is no \( (a \land \varphi_1) \) in the interval \( (t_i - l - 1, t_i - l] \). Then, there is \( k > i \), such that \( \rho'', j \models (c \land \square_{[0,1]} \neg \text{beg}_b) \) and \( \rho'', i \models (a \land \varphi_1) \). Let \( t_j \in \{t_i + l + 1\} \) be the point where \( \varphi_1 \land \varphi_2 \land \neg b \) holds. Then, we have \( \rho'', j \models (c \land \square_{[0,1]} \neg \text{beg}_b) \) and \( \rho'', i \models (a \land \varphi_1) \). Hence, \( \rho'' \models \neg \hat{X}_{[l,l+1]} \), a contradiction.

(c) Assume \( \rho'' \models \neg \varphi_{10} \). This case is similar to the case when \( \rho'' \models \neg \varphi_9 \).

So we have proved that \( \rho'' \models \psi \). Recall that \( \hat{\rho}'' = \hat{\rho}'' \downarrow_{\Sigma''} \), and \( \rho'' \) was constructed by adding oversampling points to \( \hat{\rho} \). Hence, \( \rho'' \downarrow_{\Sigma''} \hat{\rho}'' = \hat{\rho}'' \downarrow_{\Sigma''} \), giving the proof.

**Theorem 1.** For every \( \varphi \in \text{MTL}^\varphi [U_t, S_{NS}] \) over \( \Sigma \), we can construct \( \psi_{flat} \in \text{MTL}^\varphi [U_t] \) of \( \Sigma'' \supseteq \Sigma \) such that

1. For all \( \rho' \in T\Sigma'' \), if \( \rho' \models \psi \) then \( \rho' \downarrow_{\Sigma''} \models \varphi \).
2. For all \( \rho \in T\Sigma^\varphi \), if \( \rho \models \varphi \) then there exists \( \rho' \in T\Sigma'' \) such that \( \rho' \models \psi \) and \( \rho' \downarrow_{\Sigma''} = \rho \).
Appendix E illustrates in detail, the elimination of a past modality ♦. For instance, we can write aS_{l+1}b as ♦_{l+1}b \land (aSb) \land \bigwedge_{(0,l)} (a \land aSb), for r = l+1, \infty. Similarly, all intervals (l, l+1), (l, \infty) are handled. Further, S can be removed (More details can be found at Appendix G). To obtain an equisatisfiable MTL\,^w[\mathcal{U}_I] formula. Also, ♦_{[l,m]} is equivalent to ♦_{[l+1]} \lor ♦_{[l+2]} \lor \cdots \lor ♦_{[m-1,m]} Hence, the only past modalities in the formulae are ♦_{[l,i+1]} or ♦_{[l,i+1]} \lor ♦_{[l,i+1]}, Lemmas 4 and 6 show how these can be expressed in MTL\,^w[\mathcal{U}_I] to obtain equisatisfiable formulae. Hence the theorem follows.

By symmetry, using reflection 7, we can reduce MTL\,^w[\mathcal{U}_{NS}, \mathcal{S}_I] to MTL\,^w[\mathcal{U}_I, \mathcal{S}_{NS}]. Appendix E illustrates in detail, the elimination of a past modality ♦_{l+1}.a.

3.3 Expressiveness

We wind up this section with a brief discussion about the expressive powers of logics MTL\,^w[\mathcal{U}_I, \mathcal{S}_{NS}] and MTL\,^w[\mathcal{U}_{NS}, \mathcal{S}_I]. The following lemma highlights the fact that even unary modalities ♦_l, ♦_I with singular intervals are more expressive than \mathcal{U}_{NS}, \mathcal{S}_{NS}; likewise, non-singular intervals are more expressive than intervals of the form (0, \infty).

Lemma 7. (i) MTL\,^w[\cap_1] \not\subseteq MTL\,^w[\mathcal{U}_{NS}, \mathcal{S}_I], (ii) MTL\,^w[\cap, \cap_1] \not\subseteq MTL\,^w[\mathcal{U}_I, \mathcal{S}_{NS}], and (iii) MTL\,^w[\cap_{NS}, \cap_{NS}] \not\subseteq MTL\,^w[\mathcal{U}_I, \mathcal{S}].

Proof. The formula ♦_{0,1} \{a \land \neg ♦_{1,1} (a \lor b)\} in MTL\,^w[\cap_1] has no equivalent formula in MTL\,^w[\mathcal{U}_{NS}, \mathcal{S}_I]. Similarly, the formula ♦\{a \land \neg ♦_{1,1} (a \lor b)\} in MTL\,^w[\cap, \cap] has no equivalent formula in MTL\,^w[\mathcal{U}_I, \mathcal{S}_{NS}]. The formula ♦_{1,2} \{a \land \neg ♦_{1,2} a\} \subseteq MTL\,^w[\cap_{NS}, \cap_{NS}] has no equivalent formula in MTL\,^w[\mathcal{U}_I, \mathcal{S}]. A proof using EF games 13 can be seen in Appendix B.

4 Unary MTL and Undecidability

We explore the unary fragment of MTL. In this section, we show the undecidability of satisfiability checking of MTL\,^w[\cap_1, \cap] over finite timed words. The undecidability follows by construction of an appropriate MTL formula \varphi simulating a deterministic k-counter counter machine \mathcal{M} such that \varphi is satisfiable \mathcal{M} halts. We also show the non primitive recursive lower bound for satisfiability of MTL\,^w[\cap_1] by reduction of halting problem (location reachability problem) for counter machine with increment errors 16, 4 to satisfiability of the logic.

A deterministic k-counter machine is a k+1 tuple \mathcal{M} = (P, C_1, \ldots, C_k), where (i) C_1, \ldots, C_k are k-counters taking values in \mathbb{N} (their initial values are set to zero); and (ii) P is a finite set of instructions with labels p_1, \ldots, p_{n-1}, p_n. There is a unique instruction labeled HALT. For E \in \{C_1, \ldots, C_k\}, the instructions P are of the following forms: (I) p_i; Inc(E), goto p_j, (II) p_i; If E = 0, goto p_j, else go to p_k, (III) p_i; Dec(E), goto p_j, and (IV) p_i; HALT. A configuration W = (i, c_1, \ldots, c_k) of \mathcal{M} at any point of time is given by the value of the current program counter i and valuation of the counters c_1, \ldots, c_k. A move of
(error-free) counter machine \((l, c_1, \ldots, c_k) \rightarrow_{\text{std}} (l', c'_1, \ldots, c'_k)\) denotes that configuration \((l', c'_1, \ldots, c'_k)\) is obtained from \((l, c_1, \ldots, c_k)\) by executing \(l^\text{th}\) instruction. Subscript \(\text{std}\) denotes that the move is that of error-free counter machine. Let \((l^1, c_1^1, \ldots, c_k^1) \leq (l^2, c_1^2, \ldots, c_k^2)\) iff \(l^1 = l^2\) and \(\forall i \in \{1, \ldots, k\} c_i^1 \leq c_i^2\). We define a move of a counter machine with increment-errors \((l, c_1, \ldots, c_k) \rightarrow_{\text{incerr}} (l'', c''_1, \ldots, c''_k)\) iff \((l, c, d) \rightarrow_{\text{std}} (l', c'_1, \ldots, c'_k)\) and \((l', c'_1, \ldots, c'_k) \leq (l'', c''_1, \ldots, c''_k)\). Thus, machine may make increment error while moving to a next configuration.

A counter machine whose execution follows the standard moves is called Minsky Counter Machine. A counter machine whose execution follows moves with increment errors is called Incrementing Counter Machine. A computation of a counter machine (of given type) is a sequence of moves (of appropriate type) \(W_0 \rightarrow W_1 \ldots \rightarrow W_m\) where \(W_0 = (1, 0, \ldots, 0)\). The computation is terminating if the last configuration is a halting configuration, i.e. \(C_m = (n, c''_1, \ldots, c''_k)\). A counter machine is called halting if it has a terminating computation.

**Theorem 2** ([10]). Whether a given \(k\)-counter \((k \geq 2)\) Minsky machine is halting or not (equivalently the location reachability problem) is undecidable.

**Theorem 3** ([16]). Whether a given \(k\)-counter incrementing machine is halting or not (equivalently the location reachability problem) is decidable with non primitive recursive complexity.

**Encoding Minsky Machines in \(\text{MTL}^{\text{pr}}[\Diamond I, \Box I]\)**

We encode each computation of a \(k\)-counter machine \(M\) using a (non-empty set of equivalent) timed words over the alphabet \(\Sigma_M = \{b_1, b_2, \ldots, b_n, a\}\). The timed language \(L_M\) over \(\Sigma_M\) contains one and only one timed word corresponding to unique halting computation of \(M\). We then generate a formula \(\varphi_M\) such that \(L_M = L(\varphi_M)\). The encoding is done in the following way: A configuration \((i, c_1, \ldots, c_k)\) is represented by a sub-string with untimed part \(b_i a_{i} a_{i+1} \ldots a_{i+k}\). A computation of \(M\) is encoded by concatenating sequences of individual configurations. We encode the \(j^\text{th}\) configuration of \(M\) in the time interval \([(2k+1)j, (2k+1)(j+1)]\) as follows: For \(j \in \mathbb{N}\),

(i) \(b_i\) (representing instruction \(p_{i,j}\)) occurs at time \((2k+1)j\); (ii) The value of counter \(C_q\), \(q \in \{1, 2, \ldots, k\}\), in the \(j^\text{th}\) configuration is given by the number of \(a\)'s in the interval \([(2k+1)j + 2q - 1, (2k+1)j + 2q)\); (iii) The \(a\)'s can appear only in the intervals \([(2k+1)j + 2q - 1, (2k+1)j + 2q)\), \(q \in \{1, 2, \ldots, k\}\), and (iv) The intervals \([(2k+1)j + 2w, (2k+1)j + 2w + 1)\), \(w \in \{0, 1, \ldots, k\}\) have no events.

The computation must start with initial configuration and the final configuration must be the \(\text{HALT}\) instruction; beyond this, there are no more instructions. \(\varphi_M\) is obtained as a conjunction of several formulae. Let \(B\) be a shorthand for \(\bigvee_{i \in \{1, \ldots, n\}} b_i\) and let \(\text{action}\) denote \(B \lor a\). We first give some generic formulae which hold for both Minsky and Incrementing machines.

1. The symbol \(b_{ij}\) representing instruction \(p_{ij}\) occurs at \((2k+1)j\) for all \(j \in \mathbb{N}\):

\[
\varphi_0 = b_1 \land \Box\{B \land \Diamond B \Rightarrow \Diamond_B \} \land \Box\{B \Rightarrow (\neg \Diamond_{(2k+1)}B)\}.
\]
2. No events in Intervals \(((2k+1)j + 2w, (2k+1)j + 2w + 1), w \in \{0, \ldots, k\}, j \in \mathbb{N}\).

\(\varphi_1 = \Box_b B \Rightarrow \bigwedge_{w \in \{0, \ldots, k\}} (\Box_{(2w, 2w+1)}(\lnot a)).\)

3. Beyond \(p_n=\text{HALT}\), there are no instructions: \(\varphi_2 = \Box_n (b_n \Rightarrow \Box_{[2k+1, \infty)}(\text{false})\)

4. Computation starts in \(< 1, 0, \ldots, 0 >: \varphi_4 = b_1 \land \Box_{(0,2k+1)}(\text{false})\)

5. At any point of time, exactly one event takes place. Events have distinct time stamps.

\(\varphi_5 = \bigwedge_{y \in \Sigma_M}(y \Rightarrow \lnot(V_{x \in \Sigma_M \setminus \{y\}}(x)))\)

6. Eventually we reach the halting configuration \(\langle p_n, c_1, \ldots, c_k \rangle: \varphi_6 = \top b_n\)

7. We define macros, \(\text{COPY}_i, \text{INC}_i, \text{DEC}_i\) for counter \(C_i\).

- \(\text{COPY}_i\): Every \(a\) occurring in the interval \(((2k+1)j + 2i, (2k+1)j + 2i)\)
  has a copy at a future distance \(2k + 1\), and every \(a\) occurring in the next interval has an \(a\) at a past distance \(2k + 1\). This ensures the absence of insertion errors.

\(\text{COPY}_i = \Box_{(2i-1, 2i)}[(a \Rightarrow \Diamond_{[2k+1,2k+1]}(a)] \land \Box_{(2k+1+2i-1, (2k+1)+2i)}[(a \Rightarrow \Diamond_{[2k+1,2k+1]}(a)]\}

- \(\text{INC}_i\): All \(a\)'s in the current configuration are copied to the next, at a future distance \(2k + 1\); every \(a\) except the last, in the next configuration has an \(a\) at past distance \(2k + 1\).

\(\text{INC}_i = \bigwedge_{(2i-1,2i)}\{a \Rightarrow \Diamond_{[2k+1,2k+1]}(a)\}
\land [(a \land \lnot \Diamond_{[0,1]}(a)] \Rightarrow \Diamond_{[2k+1,2k+2]}(a) \land \Box_{(2k+1,2k+2)}(a \Rightarrow \Box_{[0,1]}(\lnot \text{action})))\}
\land \Box_{(2k+1+2i-1, (2k+1)+2i)}[(a \land \Diamond_{[0,1]}(a) \Rightarrow \Diamond_{[2k+1,2k+1]}(a)]\}

- \(\text{DEC}_i\): All the \(a\)'s in the current configuration, except the last, have a copy at future distance \(2k + 1\). All the \(a\)'s in the next configuration have a copy at past distance \(2k + 1\).

\(\text{DEC}_i = \Box_{(2i-1,2i)}[(a \land \Diamond_{[0,1]}(a) \Rightarrow \Diamond_{[2k+1,2k+1]}(a) \land [(a \land \lnot \Diamond_{[0,1]}(a)] \Rightarrow \lnot \Diamond_{[2k+1,2k+2]}(a)] \land \Box_{(2k+1+2i-1, (2k+1)+2i)}[(a \Rightarrow \Diamond_{[2k+1,2k+1]}(a)]\}

These macros can be used to simulate all type of instructions. We explain only the zero-check instruction here. \(p_x\): If \(C_i = 0\) goto \(p_y\), else goto \(p_z\).

\(\varphi_{3,x,0} = \Box_b b_x \Rightarrow (\bigwedge_{i \in \{1, \ldots, n\}} \text{COPY}_i \land \Box_{(2i-1,2i)}(\lnot a) \Rightarrow (\Diamond_{[2k+1,2k+1]}(b_y) \land \lnot \Diamond_{[2k+1,2k+1]}(b_z))\)

The encodings \(\varphi_{3,inc_i}, \varphi_{3,dec_i}\), corresponding to increment, decrement instructions of counter \(i\) can be found in Appendix\(\text{3}\). The final formula we construct is \(\varphi_M = \bigwedge_{i=0}^{6} \varphi_{3,i}\), where \(\varphi_3\) is the conjunction of formulae \(\varphi_{3,inc_i}, \varphi_{3,dec_i}, \varphi_{3,i=0}, i \in \{1, 2, \ldots, k\}\).

**Encoding Incrementing Counter Machines in MTL\(^{pw}\)[\(\Diamond I\)]**

To encode a computation of incrementing counter machine \(M\), we need to represent increment, decrement and no change of counter values in presence of increment errors. Since increment errors need not be checked, the encoding does not need past modality: all formulae except \(\text{COPY}_i, \text{INC}_i, \text{DEC}_i\) are in MTL\(^{pw}\)[\(\Diamond I\)]. We now give \(\text{COPY}_ERR_i, \text{INC}_ERR_i, \text{DEC}_ERR_i\) in place of
COPY, INC, DEC which allows insertion errors.

1. Copy counter with error: Copy all a’s without restricting insertions of other a’s.

\[ \text{COPYERR}_i = \Box_{(2i-1, 2i)}[(a \Rightarrow \Diamond_{[2k+1, 2k+1]a})] \]

2. Increment counter with insertion errors: Copy all a’s inserting at least 1 a after the last copied a. INCERR \[ i \] is defined as \[ \Box_{(2i-1, 2i)}\{[(a \land \Diamond_{[0, 1]}a) \Rightarrow (\Diamond_{[2k+1, 2k+1]a}] \land [(a \land \neg \Diamond_{[0, 1]}a) \Rightarrow (\Diamond_{[2k+1, 2k+1]a} \land \Diamond_{(0, 1)}a)]\} \]

3. Decrement counter with error: Copy all the a’s in an interval, except the last a

\[ \text{DECERR}_i = \Box_{(2i-1, 2i)}[(a \land \Diamond_{[0, 1]}a) \Rightarrow \Diamond_{[2k+1, 2k+1]a}] \]

Construction of \( \varphi_M \) corresponding to the incrementing counter machine is done using these macros. This is similar to the construction in Appendix F.

**Lemma 8.** Let \( M \) be a k-counter incrementing machine. Then, we can synthesize a formula \( \varphi_M \in \text{MTL}^{pw}[\Diamond I] \) such that \( M \) halts iff \( \varphi_M \) is satisfiable.

**Lemma 9.** Let \( M \) be a k-counter Minsky machine. Then, we can synthesize a formula \( \varphi_M \in \text{MTL}^{pw}[\Diamond I, \Diamond \neg I] \) such that \( M \) halts iff \( \varphi_M \) is satisfiable.

Lemma 8 and Theorem 3 together say that satisfiability of \( \text{MTL}^{pw}[\Diamond I] \) is non primitive recursive. Lemma 9 and Theorem 2 together say that satisfiability of \( \text{MTL}^{pw}[\Diamond I, \Diamond \neg I] \) is undecidable. It follows from lemma 9 that the recurrent state problem of k-counter incremental machine can also be encoded; hence \( \text{MTL}^{pw}[\Diamond I] \) is undecidable over infinite words.

## 5 Discussion

We have shown that satisfiability \( \text{MTL}^{pw}[\mathcal{U}_I, \mathcal{S}_{NS}] \) over finite strictly monotonic timed words is decidable. This subsumes the previously known decidable fragments \( \text{MTL}^{pw}[\mathcal{U}_{NS}, \mathcal{S}_{NS}] \) and \( \text{MTL}^{pw}[\mathcal{U}_I] \). The decidability is proof is carried out by extending the technique of temporal projections \[1,3,15,7\] to pointwise models in presence of oversampling. In general, this technique allows us to reduce a formula of one logic to an equi-satisfiable formulae in a different/simpler logic. It can be shown that the use of oversampling is indeed necessary to obtain equi-satisfiable formula in our reduction from \( \text{MTL}^{pw}[\mathcal{U}_I, \mathcal{S}_{NS}] \) to \( \text{MTL}^{pw}[\mathcal{U}_I, \mathcal{S}] \); we have omitted this proof for the lack of space. We believe that the technique of temporal projections with oversampling has wide applicability and it embodies an interesting notion of equivalence of formulae/logics modulo temporal projections.

In the second part of the paper, we have investigated the decidability of the unary fragment \( \text{MTL}[^1, \Diamond I] \) which is expressively weaker than full \( \text{MTL}[\mathcal{U}_I, \mathcal{S}_I] \) \[13\]. As observed by Rabinovich, the standard construction encoding a k-counter machine configuration in unit interval does not work in absence of \( \mathcal{U} \) (or \( \mathcal{S} \) operator). We have arrived at an altered encoding of a configuration using a time interval of length \( 2k + 1 \) with suitable gaps. We have shown that the restriction
of $\text{MTL}[U_I, S_I]$ to its unary fragment does not lead to any improvement in decidability. Using similar ideas, perfect channel machines can also be encoded into $\text{MTL}^{pw}[\Diamond_I, \Box_I]$ and lossy channel machines can be encoded into $\text{MTL}^{pw}[\Diamond_I]$. Our exploration has mainly looked at pointwise models with strictly monotonic time. The case of weakly monotonic time requires more work.

References

1. Rajeev Alur and Thomas A. Henzinger. Logics and Models of Real Time: A Survey. Proceedings of REX Workshop 1991, 74-106.
2. Rajeev Alur and Tomáš Feder and Thomas A. Henzinger. The Benefits of Relaxing Punctuality. Journal of the ACM, 43(1), 116–146, 1996.
3. Deepak D’Souza and M Raj Mohan and Pavithra Prabhakar. Eliminating past operators in Metric Temporal Logic. Perspectives in Concurrency, 86–106, 2008.
4. S. Demri and R. Lazic. LTL with freeze quantifier and register automata. LICS 2006, 17-26.
5. T.A. Henzinger. The Temporal Specification and Verification of Real-time Systems. Ph.D Thesis, Stanford University, 1991.
6. Y. Hirshfeld and A. Rabinovich. Logics for Real Time: Decidability and Complexity. Fundam. Inform., 62(1), 2004, 1-28.
7. D.Kini, S. N. Krishna and P. K.Pandya. On Construction of Safety Signal Automata for $\text{MTL}[U_I, S_I]$ using Temporal Projections. Proceedings of FORMATS 2011, 225-239.
8. Ron Koymans. Specifying Real-Time Properties with Metric Temporal Logic. Real Time Systems, 2(4), 255-299, 1990.
9. Oded Maler, Dejan Nickovic and Amir Pnueli. Real Time Temporal Logic: Past, Present, Future. Proceedings of FORMATS 2005, 2-16.
10. M. Minsky, Finite and infinite machines, Prentice Hall, New Jersey, 1967.
11. Joël Ouaknine and James Worrell. On the Decidability of Metric Temporal Logic. Proceedings of LICS 2005, 188–197.
12. Joël Ouaknine and James Worrell. On Metric Temporal Logic and Faulty Turing Machines. Proceedings of FOSSaCS 2006, 217–230.
13. P.K. Pandya, S. Shah. On Expressive Powers of Timed Logics: Comparing Boundedness, Non-punctuality, and Deterministic Freezing. Proceedings of CONCUR 2011, 60-75.
14. P.K.Pandya and S.Shah. The Unary Fragments of Metric Interval Temporal Logic: Bounded versus Lower Bound Constraints. Proceedings of ATVA 2012, 77-91.
15. P. Prabhakar and Deepak D’Souza. On the Expressiveness of MTL with Past Operators. Proceedings of FORMATS 2006, 322–336.
16. P. Schnoebelen. Verifying lossy channel systems has nonprimitive recursive complexity. Info. Proc. Lett. 83(5), 2002, 251-261.
Appendix

A Proof of Lemma 2

Proof. The proof idea is to apply structural induction on \( \varphi \) taking care of the new sampling points that can get added to \( \rho' \), where \( \text{act} = \bigvee \Sigma \) does not hold. The base case involves formulae of the form \( \forall \mathcal{U} \mathcal{b} \). Let \( \rho = (X_1, 0)(X_2, t_2) \ldots (X_k, t_k) \). If \( \rho, i \models \forall \mathcal{U} \mathcal{b} \), then there exists \( j > i \) where \( \mathcal{b} \) holds, and all points in between \( i, j \) satisfy \( a \). Also, \( t_j - t_i \in I \). In an oversampling \( \rho' = (Y_1, 0)(Y_2, t_2') \ldots (Y_z, t_z') \), by the definition of oversampling in section 3.1, we let \( \rho'(i + s) = i \), and let \( \rho'(j + m) = j \), \( s, m \geq 0 \). Let \( t_1, t_2, \ldots, t_{j - i} \) be points such that \( i + s < t_1 < t_2 < \cdots < t_{j - i} = j + m \) and \( \rho'(l_1) = i + 1, \rho'(l_2) = i + 2, \ldots, \rho'(l_{j - i}) = j \). Then, by the definition of oversampling in section 3.1, we have

1. \( Y_{l_d} \cap \Sigma = X_{i + d} \), for \( 1 \leq d \leq j - i \), and
2. all the sampling points \( Y_{h_y}, l_y + 1 < h_y < l_y + 1 - 1 \) are such that \( Y_{h_y} \cap \Sigma = \emptyset \)
3. \( t_{i + l} = t_i, t_{j + m} = t_j \). Thus, \( t_{i + l} - t_{i + l} = t_j - t_i \).

Hence, \( \rho, i \models a \mathcal{U} \mathcal{b} \) iff \( \rho', i + l \models (\text{act} \Rightarrow a) \mathcal{U} \mathcal{a} \mathcal{b} \). A similar result holds for past formulae. The argument for the general case follows from the base case above.

The converse can be argued in a similar way. \( \square \)

B Proof of Lemma 7

We prove that the \( \mathsf{MTL}^p_{\exists} \mathcal{U}_{\mathcal{S}_I}, \mathcal{S}_I \), \( \mathsf{MTL}^p_{\exists} \mathcal{U}_{\mathcal{I}}, \mathcal{S}_{NS} \) are strictly less expressive than \( \mathsf{MTL}^p_{\exists} \mathcal{U}_{\mathcal{I}}, \mathcal{S}_I \) using EF Games. We omit the game strategies here and give the candidate formula and pair of words.

(i) \( \mathsf{MTL}^p_{\exists} \mathcal{U}_{\mathcal{I}} \not\models \mathsf{MTL}^p_{\exists} \mathcal{U}_{\mathcal{I}}, \mathcal{S}_{NS} \)

We consider a formula in \( \mathsf{MTL}^p_{\exists} [\mathcal{U}_{\mathcal{I}}] \varphi = \hat{\varphi}_{(0, 1)} \{ a \land \neg \hat{\varphi}_{[1, 1]} (a \lor b) \} \). For an \( n \)-round game, consider the words \( w_1 = W_a W_b \) and \( w_2 = W_a W'_b \) with

- \( W_a = (a, \delta)(a, 2\delta) \ldots (a, i\delta - \kappa)(a, i\delta) \ldots (a, n\delta) \)
- \( W_b = (b, 1 + \delta)(b, 1 + 2\delta) \ldots (b, 1 + i\delta - \kappa)(b, 1 + i\delta)(b, 1 + n\delta) \)
- \( W'_b = (b, 1 + \delta)(b, 1 + 2\delta) \ldots (b, 1 + (i - 1)\delta)(b, 1 + i\delta - \kappa)(b, 1 + i\delta)(b, 1 + (i + 1)\delta)(b, 1 + n\delta) \)

\( w_1 \models \varphi \), but \( w_2 \not\models \varphi \). The key observation for duplicator’s win in an \( \mathcal{U}_{\mathcal{I}}, \mathcal{S}_I \) game is that \( a \) any non-singular future move of spoiler can be mimicked by the duplicator from \( W_a, W_b \) or \( W_a, W'_b \) (b) for any singular past move made by spoiler on \( W_a, W_b \). The same holds for any singular past move of spoiler made from \( W_a, W'_b \).

(ii) \( \mathsf{MTL}^p_{\exists} [\hat{\varphi}_{\mathcal{I}}, \mathcal{S}_I] \not\models \mathsf{MTL}^p_{\exists} [\mathcal{U}_{\mathcal{I}}, \mathcal{S}_{NS}] \)

We consider a formula in \( \mathsf{MTL}^p_{\exists} [\hat{\varphi}_{\mathcal{I}}], \phi' = \hat{\phi}_{[1, 1]} \{ a \lor \neg \hat{\phi}_{[1, 1]} (a \land b) \} \). We show that
there is no way to express this formula in $\text{MTL}^\text{pw}[\mathcal{U}_1, \mathcal{S}_{NS}]$. This is symmetrical to (i). For an $n$ round game, consider the words $w_1 = W_aW_b$ and $w_2 = W'_aW_b$

- $W_a = (a, \delta)(a, 2\delta) \cdots (a, (i-1)\delta)(a, i\delta - \kappa)(a, i\delta) \cdots (a, n\delta)$
- $W'_a = (a, \delta)(a, 2\delta) \cdots (a, (i-1)\delta)(a, i\delta) \cdots (a, n\delta)$
- $W_b = (b, 1+\delta)(b, 1+2\delta) \cdots (b, 1+(i-1)\delta)(b, 1+i\delta - \kappa)(b, 1+i\delta) \cdots (b, 1+n\delta)$

$w_1 \not\models \varphi'$, $w_2 \models \varphi'$. The key observation for duplicator’s win in an $n$-round $\mathcal{U}_1, \mathcal{S}_{NS}$ game is that (a) any non-singular past move by spoiler from $W_a, W_b$ or from $W'_a, W_b$ can be answered by duplicator, (b) for any singular future move made by spoiler on $W_a, W_b$, duplicator has a reply from $W'_a, W_b$. The same holds for any singular future move of spoiler made from $W'_a, W_b$.

(iii) $\text{MTL}^\text{pw}[\hat{\Phi}_{NS}, \hat{\Phi}_{NS}] \not\subseteq \text{MTL}^\text{pw}[\mathcal{U}_1, \mathcal{S}]$. We consider the $\text{MTL}^\text{pw}[\hat{\Phi}_{NS}, \hat{\Phi}_{NS}]$ formula $\varphi'' = \hat{\Phi}_{(1,2)}[a \land \neg \Phi_{(1,2)}]$, and show that there is no way to express it using $\mathcal{U}_1, \mathcal{S}$. For an $n$ round game, consider the words $w_1 = W_1W_2$ and $w_2 = W_1W'_2$ with

- $W_1 = (a, 0.5 + \epsilon) \cdots (a, 0.5 + ne)(a, 0.9 + \epsilon) \cdots (a, 0.9 + ne)$
- $W_2 = (a, 1.5)(a, 1.6 + \epsilon)(a, 1.6 + 2\epsilon) \cdots (a, 1.6 + ne)$
- $W'_2 = (a, 1.6 + \epsilon)(a, 1.6 + 2\epsilon) \cdots (a, 1.6 + ne)$

for a very small $\epsilon > 0$. Clearly, $w_1 \models \varphi'', w_2 \not\models \varphi''$. The key observation for duplicator’s win in an $n$-round $\mathcal{U}_1, \mathcal{S}$ game is that (a) when spoiler picks any position in $W_1$, duplicator can play copy cat, (b) when spoiler picks $(a, 1.5)$ in $W_2$ as part of a future $(0, 1)$ move from $W_1$, duplicator picks $0.9 + ne$ in $W'_2$. All until, since moves from the configuration $[(a,1.5), (a,0.9+ne)]$ are symmetric.

C Converse of Lemma 4

Conversely, assume $\rho' \models \nu$. Let $\rho' \models \neg \hat{X}_{[l,\infty)}$. Then, there is a point $i$ such that $\rho', i \models \text{act}$ and $\rho', i \not\models (b \leftrightarrow \hat{\varphi}_{[l,\infty)}(a \land \text{act}))$. Assume that $\rho', i \models b$, but $\rho', i \not\models \hat{\varphi}_{[l,\infty)}(a \land \text{act})$. Then, all points $\text{act}$ in $[0, t_i - l]$ are marked $\neg a$. Then, by $\varphi_1$,

1. all points $\text{act}$ in the $[0, l)$ future of the first $a \land \text{act}$ must be marked $\neg b$
2. $\neg b \land \neg a$ holds at all points $\text{act}$ till the first $a \land \text{act}$.

Given the above two points, we cannot have a $b$ at $t_i$. Thus, $\rho', i \not\models \hat{\varphi}_{[l,\infty)}(a \land \text{act})$, and $\rho', i \models \text{act}$ gives $\rho', i \models \neg b$. We can similarly show that if $\rho', i \models \neg b \land \text{act}$, then $\rho', i \models \hat{\varphi}_{[l,\infty)}(a \land \text{act})$.

D Eliminating $\mathcal{S}$

Given a formula $\varphi \in \text{MTL}^\text{pw}[\mathcal{U}_1, \mathcal{S}]$ over $\Sigma$, we first flatten the formula to obtain an equisatisfiable formula $\varphi_{\text{flat}}$ over $\Sigma' \supset \Sigma$. In this section, we elaborate [7].
on removing the temporal definitions of the form \([r \leftrightarrow (c \, S \, f)]\) from \(\varphi_{flat}\), using future operators. We use the shortform \(\mathcal{O}_{\varphi}\) to denote \(false \mathcal{U} \varphi\).

\([r \leftrightarrow (c \, S \, f)]\) will be replaced by a conjunction \(\nu_r\) of the following future formulae:

- \(\varphi_1 : \lozenge(f \Rightarrow \mathcal{O} \, r)\)
- \(\varphi_2 : \neg \nu_r\)
- \(\varphi_3 : \lozenge((r \land c) \Rightarrow \mathcal{O} \, r)\)
- \(\varphi_4 : \lozenge (r \land (\neg c \land \neg f)) \Rightarrow \mathcal{O} \neg \nu_r\)
- \(\varphi_5 : \lozenge ((\neg r \land \neg f) \Rightarrow \mathcal{O} \neg \nu_r)\)

For example, consider the formula \(\varphi = (a \land (b \land (c \, U_{(1,2)}[d \, S \, e] \land f)))\). The flattened version \(\varphi_{\text{flat}} = [(d \, S \, e) \iff w_1] \land [w_2 \iff c \, U_{(1,2)}[w_1 \land f]] \land (a \land b \land w_2)\).

Replace \([(d \, S \, e) \iff w_1] with \(\nu_{w_1}\) to obtain the equisatisfiable formula:

\[\lozenge(e \Rightarrow \mathcal{O} \, w_1) \land \neg \nu_1 \land \square((w_1 \land d) \Rightarrow \mathcal{O} \, w_1) \land [\square(w_1 \land (\neg d \land \neg e) \Rightarrow \mathcal{O} \, \neg w_1] \land \square((\neg w_1 \land \neg e) \Rightarrow \mathcal{O} \, \neg w_1] \land [w_2 \iff c \, U_{(1,2)}[w_1 \land f]] \land (a \land b \land w_2).\]

E An Example

We give a toy example for elimination of past operator using technique described above. Consider the formula \(\varphi = (b \Rightarrow \lozenge_{(1,2)}a) \mathcal{U}(EP)\) over \(\Sigma = \{a,b\}\). Formula says that before the word ends, wherever there is a \(b\) there must be an \(a\) in its past with time difference in \((1,2)\). We now eliminate \(P_{(1,2)}\) as follows:

- **Flattening:** We construct a \(\hat{\varphi}_{\text{flat}}\) over \(\Sigma = \{a,b\}\) which is equi-satisfiable to our original formula (implied by \(\Box\)). \(\hat{\varphi}_{\text{flat}} = [(b \Rightarrow (w \land act)) \mathcal{U}(EP \land act) \land X. X = \Diamond (act \Rightarrow (w \leftrightarrow (\lozenge_{(1,2)}a \land act)))\). Note that The formula asserts that at any old action point wherever \(\lozenge_{(1,2)}a\) is true, it is marked as \(w\). Thus \(w \land \neg \nu_r\) acts as a witness for \(\lozenge_{(1,2)}a\) and further wherever there is a \(b\) that point should be marked as \(w \) which means that all the points where \(b\) holds \(\lozenge_{(1,2)}a\) also holds.

- **Canonicalization:** We construct a canonical language \(L'\) over \(\Sigma_2 = \Sigma \cup \{c, \, \text{end}_w, \, \text{beg}_w\}\) that for any word \(\rho \in L(\hat{\varphi}_{\text{flat}})\) there is a unique \(\rho' \in L'\) such that \(L' \models_{\Sigma_2} \hat{\varphi}_{\text{flat}}\) and:

  - All integral time points of \(\rho'\) is marked as \(c\) till the end of the word and no other points are marked as \(c\).

    \(C_1 = c \land \lozenge(c \Rightarrow c \land \square((c \land \neg EP) \Rightarrow \square(0,1) \land (\lozenge_{(1,1)}c \lor \lozenge_{(0,1)}EP))\)

  - From every first occurrence of \(a\) in an interval of the form \([x, x+1)\) where \(x \in I_+ \cup \{0\}\) after exactly \(l\) time units, \(\text{end}_w\) holds.

    \(C_2 = \square(c \land \lozenge_{(0,1)}(a \land act)) \Rightarrow [(act \Rightarrow \neg a) \mathcal{U}_{(0,1)}[(a \land act) \land \lozenge_{(1,1)} \land (\lozenge_{(0,1)} \land ep)]] \land \square(0,1) \land c \land \text{end}_w\)

  - From every last occurrence of \(a\) in an interval of the form \([x, x+1)\) where \(x \in I_+ \cup \{0\}\) after exactly \(l+1\) time units, \(\text{beg}_w\) holds.

    \(C_3 = \square(c \land \lozenge_{(0,1)}(a \land act)) \Rightarrow \lozenge_{(0,1)}[(a \land act) \land [(act \Rightarrow \neg a) \land \neg c] \mathcal{U} e] \land \lozenge_{(2,1)} \lor \lozenge_{(0,2)}(\mathcal{E} P)] \land \square(0,2) \land \neg \text{beg}_w\)
Note that off the consecutive configuration. To avoid such insertion errors, we make use of

\[ C_4 = c \land \Box_{[0,1]}(\text{act} \Rightarrow \neg a) \Rightarrow \Box_{[1,2]} \neg \text{beg}_b \land \Box_{[2,3]} \neg \text{beg}_b \land c \land \tilde{\Box}_{[0,1]}x \Rightarrow (\neg x \bar{U}_{[0,1]}[x \land (\neg x \land \neg c)](c \lor EP)) \] for \( x \in \{\text{end}_b, \text{beg}_b\}\)

All \( \rho' \) which cannot be described as above is not in the \( L' \).

- **Eliminating Past Operator**: In this step we eliminate past operator by constructing a formula \( \psi \) which when replaces \( X \), results in the same timed language. Note that \( X \) exactly identifies which old action points should be marked as \( w \) and which should not be marked as \( w \).

• **Marking points as \( w \)**: According to \( X \) all those old action points having \( a \) in their past \((1,2)\) should be marked as \( w \). Which means that from any point which is marked \( a \) all the old action points should be marked \( w \).

\( Y_1 = \Box[(a \land \text{act}) \Rightarrow \Box_{[1,2]}(\text{act} \Rightarrow b)] \)

• **Avoiding all other points to be marked as \( b \)** Here we restrict the marking of \( b \) to only those points which are marked by \( Y_1 \). By main lemma we give following formulas, which restric this behavior. In brief, all those points from \( \text{beg}_b \) to \( \text{end}_b \) of the same interval should not be marked as \( b \).

\( x = (\neg (\text{end}_b \land c \land EP)) \)

\[ Y_2 = \Box[(c \land \Box_{[0,1]} \neg \text{beg}_b) \Rightarrow ((\text{act} \Rightarrow \neg b) \land x) \bar{U}(\text{end}_b \lor c \lor EP)] \land \Box[(c \land [\neg \text{end}_b \U_{[0,1]} \text{beg}_b)] \Rightarrow \Box_{[0,1]}((\text{beg}_b \land ((\text{act} \Rightarrow \neg b) \land x)U(\text{end}_b \lor c \lor EP))) \] \]

The temporal projection \( X \) can be replaced by \( \psi = C_1 \land C_2 \land C_3 \land C_4 \land Y_1 \land Y_2 \).

Note that \( \psi \) is pure \( \text{MTL}^{pw}[U] \) formula such that \( \delta = (\text{act} \Rightarrow (b \Rightarrow (w \land \text{act})))U(EP \land \text{act}) \land \psi \) equisatisfiable to \( \varphi \).

**F Lemma [9]: All Formulae**

The main challenge in encoding Minsky Machine is to avoid insertion errors in the consecutive configuration. To avoid such insertion errors, we make use of the \( \tilde{\Box} \) operator. We make use of the shorthand \( B \) for \( \bigvee_{i \in 1, \ldots, n} b_i \) and \( \text{action} \) for \( \text{true} \).

1. Insertion of \( b_j \) exactly at points \((2k+1)j, j \in \mathbb{N}\):

\[ \varphi_0 = b_1 \land \Box \{B \land \diamond B \Rightarrow \diamond_{[2k+1,2k+1]}B\} \land \Box \{B \Rightarrow (\neg \diamond_{[0,2k+1]}B)\} \]

2. Intervals with no \( a \): Intervals \((2k+1)j + 2w, (2k+1)j + 2w + 1\) have no \( a \)'s for \( 0 \leq w \leq k \).

\[ \varphi_1 = \Box \{B \Rightarrow \bigwedge_{w \in [0, \ldots, k]} \Box_{[2w,2w+1]}(\neg a)\}. \]

We define macros for copying, incrementing and decrementing counters.

- **COPY**: Every \( a \) occurring in the current interval has a copy at a future distance \( 2k+1 \), and every \( a \) occurring in the next interval has an \( a \) at a past distance \( 2k+1 \). This ensures the absence of insertion errors.
The final formula we construct is \( \varphi \) over timed state sequences.

Continuous Semantics:
Continuous time MTL formulae are typically evaluated over timed state sequences (TSS), where a system is assumed to continue in a state until a state change takes place. Here we interpret over timed words, \( \varphi \) in a state until a state change takes place. Here we interpret over timed words,

\[ G \text{ Syntax and Semantics of } a \]

Here is that the atomic formula \( \varphi \) but now a formulae can be asserted at any arbitrary time point. A small change with \( \varphi \) with \( \varphi \) and not for an interval of time after the event happens. Given a

\[ \text{COPY}_i = \Box_{(2i-1,2i)}[(a \Rightarrow \Diamond [2k+1,2k+1+1][a]) \land \Box_{(2k+1+2i-1,2k+1+1+2i)}[(a \Rightarrow \Diamond [2k+1,2k+1+1][a])]
\]

\[ \text{INC}_i: \text{All the } a \text{'s in the current configuration are copied to the next configuration, at a future distance } 2k+1; \text{ every } a \text{ except the last one in the next configuration has an } a \text{ at past distance } 2k+1.
\]

\[ \text{INC}_i = (\Box_{(2i-1,2i)}[a \Rightarrow \Diamond [2k+1,2k+1+1][a]) \land [(a \land \Diamond [0,1][a]) \Rightarrow \Diamond [2k+1,2k+1+1+2i][a]] \land \Box_{(2k+1+2i-1,2k+1+1+2i)}[(a \Rightarrow \Diamond [0,1][a]) \Rightarrow \Diamond [2k+1,2k+1+1][a])]
\]

\[ \text{DEC}_i: \text{All the } a \text{'s in the current configuration, except the last one, have a copy at future distance } 2k+1. \text{ All the } a \text{'s in the next configuration have a copy at past distance } 2k+1.
\]

\[ \text{DEC}_i = \Box_{(2i-1,2i)}[(a \land \Diamond [0,1][a]) \Rightarrow \Diamond [2k+1,2k+1+1][a]) \land [(a \land \neg F_{[0,1]}[a]) \Rightarrow \neg F_{[2k+1,2k+1+1+2i][a]} \land \Box_{(2k+1+2i-1,2k+1+1+2i)}[(a \Rightarrow \Diamond [2k+1,2k+1+1][a])]
\]

Using the macros, we define formulae for increment, decrement and conditional jumps.

\[ p_2: \text{If } C_i = 0 \text{ goto } p_y, \text{ else goto } p_z
\]

\[ \varphi_3^{x,y} = \Box_{b_x \Rightarrow (\land j \in \{1, \ldots, n\} \text{ COPY}_i \land \Box_{[2i-1,2i]}[(a \Rightarrow \Diamond [2k+1,2k+1+1][b_y]) \land \Diamond [0,1][a]) \Rightarrow (a \Rightarrow \Diamond [2k+1,2k+1+1][b_y])]
\]

\[ p_x: \text{Inc}(C_i) \text{ goto } p_y
\]

\[ \varphi_3^{x,\text{inc}_i} = \Box_{b_x \Rightarrow (\land j \in \{1, \ldots, n\} \text{ COPY}_i \land \Box_{[2i-1,2i]}[(a \Rightarrow \Diamond [2k+1,2k+1+1][b_y]) \land \Box_{[2k+1,2k+1+1+2i]}[(a \Rightarrow \Diamond [2k+1,2k+1+1][b_y])]
\]

\[ p_x: \text{Dec}(C_i) \text{ goto } p_y
\]

\[ \varphi_3^{x,\text{dec}_i} = \Box_{b_x \Rightarrow (\land j \in \{1, \ldots, n\} \text{ COPY}_i \land \Box_{[2i-1,2i]}[(a \Rightarrow \Diamond [2k+1,2k+1+1][b_y]) \land \Box_{[2k+1,2k+1+1+2i]}[(a \Rightarrow \Diamond [2k+1,2k+1+1][b_y])]
\]

\[ 4. \text{ No instructions are executed after HALT:}
\]

\[ \varphi_2 = \Box_{b_n \Rightarrow \Box_{[2k+1,\infty]}(\text{false})}
\]

\[ 5. \text{ Initial Configuration:}
\]

\[ \varphi_4 = b_1 \land \Box_{[0,2k+1]}(\text{false})
\]

\[ 6. \text{ Mutual Exclusion:- At any point of time, exactly one event takes place.}
\]

\[ \varphi_5 = \land_{y \in \Sigma} (y \Rightarrow \neg \land_{x \in \Sigma} (x))
\]

\[ 7. \text{ Termination: The HALT instruction will be seen sometime in the future.}
\]

\[ \varphi_6 = \Diamond b_n
\]

The final formula we construct is \( \varphi_M = \land_{i=0}^6 \varphi_i \), where \( \varphi_3 \) is the conjunction of formulae \( \varphi_3^{inc}, \varphi_3^{dec}, \varphi_3 = 0, i \in \{1, 2, \ldots, k\} \).

G Syntax and Semantics of MTLc

Continuous Semantics: Continuous time MTL formulae are typically evaluated over timed state sequences (TSS), where a system is assumed to continue in a state until a state change takes place. Here we interpret over timed words, but now a formulae can be asserted at any arbitrary time point. A small change here is that the atomic formula \( a \in \Sigma \) can only hold at an action point labeled \( A \) with \( a \in A \) and not for an interval of time after the event happens. Given a
timed word σ, and an MTL formula ϕ, in the continuous semantics, the temporal
connectives of ϕ quantify over the whole time domain \( \mathbb{R}_{\geq 0} \).

For an alphabet Σ, a timed word \( ρ = (σ, τ) = (A_1, t_1) \ldots (A_n, t_n) \), a time
\( t \in \mathbb{R}_{\geq 0} \), and an MTL formula ϕ, the satisfaction of ϕ at time t of ρ is denoted
\( (ρ, t) \models ϕ \), and is defined as follows: Let \( ρ(t_i) = A_i \).

\[
\begin{align*}
ρ, t &\models a \leftrightarrow a \in ρ(t) \\
ρ, t &\not\models ϕ \leftrightarrow ρ, t \not\models ϕ \\
ρ, t &\models ϕ \land ψ \leftrightarrow ρ, t \models ϕ \land ρ, t \models ψ \\
ρ, t &\models ϕ \lor ψ \leftrightarrow ρ, t \models ϕ \lor ρ, t \models ψ \\
ρ, t &\models ϕ \rightarrow ψ \leftrightarrow \neg ϕ \lor ρ, t \models ψ \\
ρ, t &\models ϕ \leftrightarrow ψ \leftrightarrow (ρ, t \models ϕ \leftrightarrow ρ, t \models ψ) \\
ρ, t &\models ϕ \land ψ \leftrightarrow ρ, t \models ϕ \land ρ, t \models ψ \\
ρ, t &\models ϕ \rightarrow ϕ \leftrightarrow ρ, t \models ϕ \\
ρ, t &\models ϕ \lor ϕ \leftrightarrow ρ, t \models ϕ \\
ρ, t &\models ϕ \rightarrow ϕ \land ϕ \leftrightarrow ρ, t \models ϕ \land ρ, t \models ϕ
\end{align*}
\]

We say that \( ρ, \emptyset \models ϕ \) iff \( ρ, 0 \models ϕ \).

\( L(ϕ) = \{ ρ \mid ρ, 0 \models ϕ \} \).

G.1 Undecidability of Continuous time Logic MTL^c[♦I]

In this section, we show undecidability of MTL^c[♦I], by encoding 2 counter machines.

1. Copy counter exactly: both action and non-action points are copied.

\[
COPY^c_C = \Box_{[1,2]}[(\neg action) \Rightarrow \Diamond_{[5,5]}(\neg action) \land (a \Rightarrow \Diamond_{[5,5]}a)]
\]

2. Increment counter by one exactly:

\[
INC^c_C = \Box_{[1,2]} \{(\neg action \land \Diamond_{[0,1]}a) \Rightarrow (\Diamond_{[5,5]}(\neg action)) \land (a \land \Diamond_{[0,1]}a) \Rightarrow (\Diamond_{[5,5]}a) \land (a \land \neg \Diamond_{[0,1]}a) \Rightarrow (\Diamond_{[5,5]}a \land \Diamond_{[5,6]}a \land \Box_{[5,6]}a \Rightarrow \Box_{[0,1]}(\neg action))\}
\]

3. Decrement counter by one exactly:

\[
DEC^c_C = \Box_{[1,2]} \{(\neg action \land \Diamond_{[0,1]}a) \Rightarrow \Diamond_{[5,5]}(\neg action) \land (a \land \Diamond_{[0,1]}a) \Rightarrow \Diamond_{[5,5]}a \land (a \land \neg \Diamond_{[0,1]}a) \Rightarrow (\Box_{[3,6]}(\neg action))\}
\]

The other formulae needed can be obtained from section 4 with \( k = 2 \).

Let the resulting version of formula be called \( \varphi''_{M} \).

Lemma 10. Let \( M \) be a two counter Minsky machine and let \( \varphi''_{M} \in \text{MTL}^c[♦I] \)
be the formula as above. Then, \( M \) halts iff \( \varphi''_{M} \) is satisfiable.

The proof of equivalence of \( L(M) \) and \( L(\varphi''_{M}) \) is similar to Lemma 9.