Metric Temporal Logic With Counting

S.N.Krishna, Khushraj Madnani, Paritosh.K.Pandya

February 1, 2016

Introduction

- *Metric Temporal Logic* is extensively studied Real time Logic in the literature.

Introduction

- *Metric Temporal Logic* is extensively studied Real time Logic in the literature.
- Allows timing constraints to be specified along with the temporal ordering.

Introduction

- *Metric Temporal Logic* is extensively studied Real time Logic in the literature.
- Allows timing constraints to be specified along with the temporal ordering.
- Exhibits considerable diversity in expressiveness and decidability properties based on restriction on modalities and type of timing constraints.

Introduction

- *Metric Temporal Logic* is extensively studied Real time Logic in the literature.
- Allows timing constraints to be specified along with the temporal ordering.
- Exhibits considerable diversity in expressiveness and decidability properties based on restriction on modalities and type of timing constraints.
- In general satisfiability checking for MTL is undecidable.

Introduction

- *Metric Temporal Logic* is extensively studied Real time Logic in the literature.
- Allows timing constraints to be specified along with the temporal ordering.
- Exhibits considerable diversity in expressiveness and decidability properties based on restriction on modalities and type of timing constraints.
- In general satisfiability checking for MTL is undecidable.
- Counting within a given time slot is a very natural and useful property in real time systems.

Introduction

- *Metric Temporal Logic* is extensively studied Real time Logic in the literature.
- Allows timing constraints to be specified along with the temporal ordering.
- Exhibits considerable diversity in expressiveness and decidability properties based on restriction on modalities and type of timing constraints.
- In general satisfiability checking for MTL is undecidable.
- Counting within a given time slot is a very natural and useful property in real time systems.
- Thus it becomes interesting to study satisfiability checking for its fragments and their extensions with ability to count.

Presentation Flow

- Model : Timed Words
- Timed Logic with Counting : Syntax and Semantics
- Temporal Projections : Simple and Oversampled
- Expressiveness Relations with Counting Extensions
- Satisfiability Checking: Decidability
- Conclusion
- Future Work

Model : Timed Word

Model : Timed Word

- Models over which pointwise MTL Formula is being evaluated

  .

Model : Timed Word

- Models over which pointwise MTL Formula is being evaluated
  .
- Finite sequence of symbols along with their corresponding timestamps.

Model : Timed Word

- Models over which pointwise MTL Formula is being evaluated
.
- Finite sequence of symbols along with their corresponding timestamps.In general, timestamps monotonically increases

Model : Timed Word

- Models over which pointwise MTL Formula is being evaluated .
- Finite sequence of symbols along with their corresponding timestamps.In general, timestamps monotonically increases
- For the purpose of this presentation we will restrict our timed words to be strictly monotonic.

Model : Timed Word

- Models over which pointwise MTL Formula is being evaluated
  .
- Finite sequence of symbols along with their corresponding timestamps.In general, timestamps monotonically increases
- For the purpose of this presentation we will restrict our timed words to be strictly monotonic.

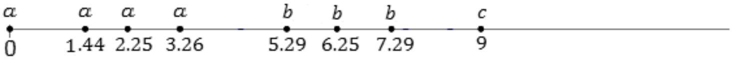


Figure: A finite timed word over $\Sigma = \{a, b, c\}$. A strictly monotonic timed word can be seen as a real line annotated with symbols from Σ

Presentation Flow

- Model : Timed Words
- Timed Logic with Counting : Syntax and Semantics
- Temporal Projections : Simple and Oversampled
- Expressiveness Relations with Counting Extensions
- Satisfiability Checking: Decidability
- Conclusion
- Future Work

- MTL Syntax

- MTL Syntax

$$\phi ::= AP \mid \phi \wedge \phi \mid \phi \vee \phi \mid \neg \phi \mid \phi \ \mathsf{U}_I \ \phi \mid \phi \ \mathsf{S}_I \ \phi$$

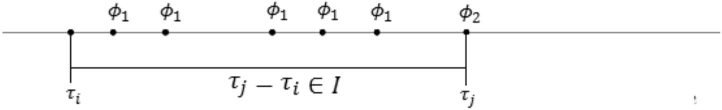
where I is interval of the form $\langle x, y \rangle$, $x \in \mathcal{N} \cup \{0\}$, $y, x \in \mathcal{N} \cup \{0, \infty\}$ and $\langle ... \rangle \in \{[...], (...), [...), (...]\}$

Metric Temporal Logic : Semantics

MTL Semantics

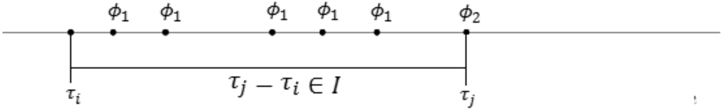
MTL Semantics

$\rho, i \models \phi_1 \, \mathsf{U}_I \phi_2 \iff \exists j > i \;\, \rho, j \models \phi_2$ *and* $\tau_j - \tau_i \in I$ *and* $\forall\, i < k < j \;\, \rho, k \models \phi_1$

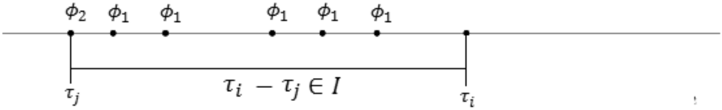


MTL Semantics

$\rho, i \models \phi_1 \, \mathsf{U}_I \phi_2 \iff \exists j > i \;\; \rho, j \models \phi_2$ *and* $\tau_j - \tau_i \in I$ *and* $\forall\, i < k < j \;\; \rho, k \models \phi_1$

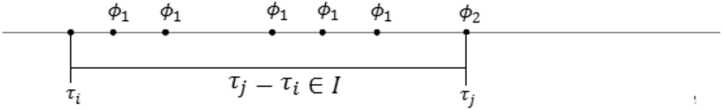


$\rho, i \models \phi_1 \, \mathsf{S}_I \phi_2 \iff \exists j < i \;\; \rho, j \models \phi_2$ *and* $\tau_i - \tau_j \in I$ *and* $\forall\, i > k > j \;\; \rho, k \models \phi_1$

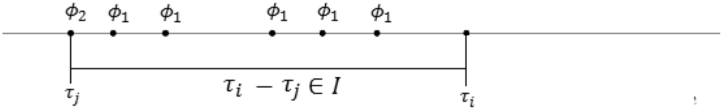


MTL Semantics

$\rho, i \models \phi_1 \, \mathsf{U}_I \phi_2 \iff \exists j > i \ \rho, j \models \phi_2$ *and* $\tau_j - \tau_i \in I$ *and* $\forall \ i < k < j \ \rho, k \models \phi_1$



$\rho, i \models \phi_1 \, \mathsf{S}_I \phi_2 \iff \exists j < i \ \rho, j \models \phi_2$ *and* $\tau_i - \tau_j \in I$ *and* $\forall \ i > k > j \ \rho, k \models \phi_1$



Ex: work-hard $\mathsf{U}_{[5,10]}$ giving-up

Subclasses:

Subclasses:

- By restricting set of allowed intervals. e.g. $\mathsf{MTL}[\mathsf{U}_{\mathsf{np}}, \mathsf{S}_{\mathsf{np}}]$, where *np* refers to non-punctual intervals. It is well known as MITL in the literature.

Time Logic : MTL Fragments

Subclasses:

- By restricting set of allowed intervals. e.g. $\mathsf{MTL}[\mathsf{U}_{\mathsf{np}}, \mathsf{S}_{\mathsf{np}}]$, where *np* refers to non-punctual intervals. It is well known as MITL in the literature.
- By restricting set of operators. We denote MTL[W] for subclass of MTL restricted to operators in W. e.g. $\mathsf{MTL}[\,\mathsf{U}_{\mathsf{I}}]$ where only until operator is allowed.

Subclasses:

- By restricting set of allowed intervals. e.g. $\mathrm{MTL}[\mathrm{U}_{\mathrm{np}}, \mathrm{S}_{\mathrm{np}}]$, where *np* refers to non-punctual intervals. It is well known as MITL in the literature.
- By restricting set of operators. We denote MTL[W] for subclass of MTL restricted to operators in W. e.g. $\mathrm{MTL}[\,\mathrm{U}_{\mathrm{I}}]$ where only until operator is allowed.
- We will restrict to future only fragment of MTL.

Time Logic with counting: CTMTL

- We introduce two new modal operators for counting C and UT.

Time Logic with counting: CTMTL

- We introduce two new modal operators for counting C and UT.
- CTMTL Syntax

$$\phi \ ::= \ AP \mid \phi \ \wedge \ \phi \mid \phi \ \vee \ \phi \mid \neg \ \phi \mid \phi \ \mathsf{U}_{I,\#\phi\sim n} \ \phi \mid \mathsf{C}_I^n\phi$$

where I is interval of the form $\langle x, y\rangle$, $x \in \mathcal{N} \cup \{0\}$,
$y, x \in \mathcal{N} \cup \{0, \infty\}$,
$\langle ...\rangle \in \{[...], (...), [...), (...]\}$,

Time Logic with counting: CTMTL

- We introduce two new modal operators for counting C and UT.
- CTMTL Syntax

$$\phi \ ::= \ AP \mid \phi \ \wedge \ \phi \mid \phi \ \vee \ \phi \mid \neg \ \phi \mid \phi \ \mathsf{U}_{I,\#\phi\sim n} \ \phi \mid \mathsf{C}_I^n \phi$$

where I is interval of the form $\langle x, y\rangle$, $x \in \mathcal{N} \cup \{0\}$,
$y, x \in \mathcal{N} \cup \{0, \infty\}$,
$\langle ...\rangle \in \{[...], (...), [...), (...]\}$,
$\sim = \{\geq, \leq\}$ and

Time Logic with counting: CTMTL

- We introduce two new modal operators for counting C and UT.
- CTMTL Syntax

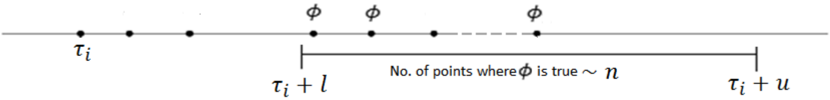
$$\phi \ ::= \ AP \mid \phi \ \wedge \ \phi \mid \phi \ \vee \ \phi \mid \neg \ \phi \mid \phi \ \mathsf{U}_{I,\#\phi\sim n} \ \phi \mid \mathsf{C}_I^n \phi$$

where I is interval of the form $\langle x, y \rangle$, $x \in \mathcal{N} \cup \{0\}$,
$y, x \in \mathcal{N} \cup \{0, \infty\}$,
$\langle ... \rangle \in \{[...], (...), [...), (...]\}$,
$\sim = \{\geq, \leq\}$ and
$n \in \mathcal{N} \cup \{0\}$

CTMTL: Semantics

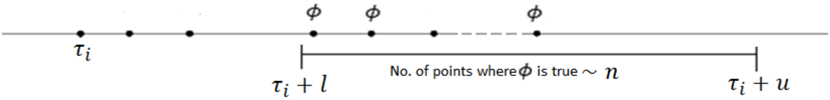
CTMTL: Semantics

$$\rho, i \models C^{\sim n}_{\langle l,u \rangle}\phi \iff N_\phi(\langle \tau_i + l, \tau_i + u\rangle) \sim n$$

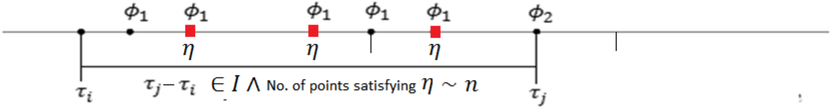


CTMTL: Semantics

$$\rho, i \models C^{\sim n}_{\langle l,u\rangle}\phi \iff N_\phi(\langle \tau_i + l, \tau_i + u\rangle) \sim n$$



$\rho, i \models \phi_1 \mathsf{U}_{I,\#\eta\sim n}\phi_2 \iff \exists j > i\ \rho, j \models \phi_2 \wedge \tau_j - \tau_i \in$
$I \wedge \forall \wedge i < k < j\ \rho, k \models \phi_1 \wedge \mathcal{N}'_\phi(i,j) \sim n$



Subclasses of CTMTL

- $\mathrm{C}_{(0,1)}$MTL: Counting of the form $C^{\sim n}_{(0,1)}$.

Subclasses of CTMTL

- $\mathrm{C}_{(0,1)}$MTL: Counting of the form $C^{\sim n}_{(0,1)}$.
- $\mathrm{C}_{(0,u)}$MTL: Counting of the form $C^{\sim n}_{(0,u)}$.

Subclasses of CTMTL

- $C_{(0,1)}$MTL: Counting of the form $C^{\sim n}_{(0,1)}$.
- $C_{(0,u)}$MTL: Counting of the form $C^{\sim n}_{(0,u)}$.
- CMTL: Counting with C modality only.

Subclasses of CTMTL

- $\mathrm{C}_{(0,1)}$MTL: Counting of the form $C^{\sim n}_{(0,1)}$.
- $\mathrm{C}_{(0,u)}$MTL: Counting of the form $C^{\sim n}_{(0,u)}$.
- CMTL: Counting with C modality only.
- TMTL: Counting with UT Modality only.

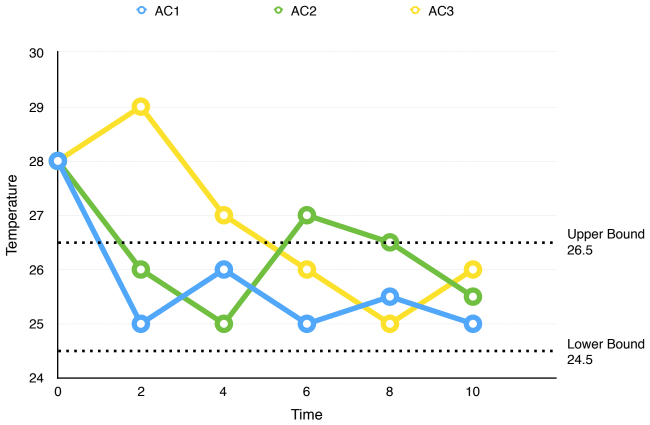
Scheduling HVAC in Demand Response: An Example

- In Demand Response system an important requirement is to reduce the Peak Power Demand.

Scheduling HVAC in Demand Response: An Example

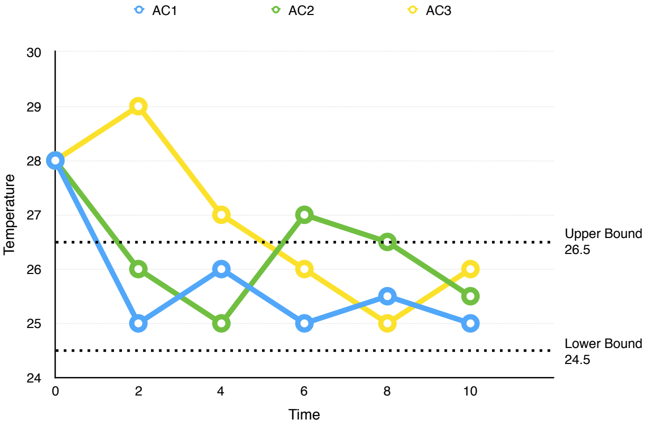
- In Demand Response system an important requirement is to reduce the Peak Power Demand.
- Scheduling of HVAC to limit peak power demand below threshold.
- HVAC are more flexible as compared to devices like microwave oven.
- Constant mode switching (OFF-¿ON) causes wear and tear and more power consumption due to transient currents.
- No. of Switch yet another important parameter to grade such scheduling algorithms.

Scheduling HVAC in Demand Response: An Example



Scheduling HVAC in Demand Response: An Example

$\Diamond_{(0,3),\#Switch-ON-AC\leq 3}(Comfort - AC_1 \wedge Comfort - AC_2 \wedge Comfort - AC_3)$



Presentation Flow

- Model : Timed Words
- Timed Logic with Counting : Syntax and Semantics
- Temporal Projections : Simple and Oversampled
- Expressiveness Relations with Counting Extensions
- Decidability : Satisfiability Checking
- Conclusion
- Future Work

Definitions: Simple Projection

Definitions: Simple Projection

- Let Σ, X be finite disjoint set.

Definitions: Simple Projection

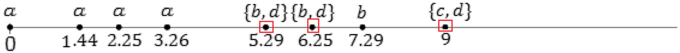
- Let Σ, X be finite disjoint set.
- **Simple Extension** A (Σ, X)-simple extension is a timed word ρ over $2^X \cup \Sigma$ such that at any point $i \in dom(\rho)$, $\sigma_i \cap \Sigma \neq \emptyset$

Definitions: Simple Projection

- Let Σ, X be finite disjoint set.
- **Simple Extension** A (Σ, X)-simple extension is a timed word ρ over $2^X \cup \Sigma$ such that at any point $i \in dom(\rho)$, $\sigma_i \cap \Sigma \neq \emptyset$
- **Simple Projection** A timed word ρ over Σ obtained by deleting symbols in X from (Σ, X) extension ρ' is called its Simple Projection.

Definitions: Simple Projection

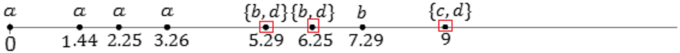
- Let Σ, X be finite disjoint set.
- **Simple Extension** A (Σ, X)-simple extension is a timed word ρ over $2^X \cup \Sigma$ such that at any point $i \in dom(\rho)$, $\sigma_i \cap \Sigma \neq \emptyset$
- **Simple Projection** A timed word ρ over Σ obtained by deleting symbols in X from (Σ, X) extension ρ' is called its Simple Projection.



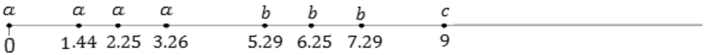
A valid Σ, X Extension $\Sigma = \{a, b, c\}$ $X = \{d\}$

Definitions: Simple Projection

- Let Σ, X be finite disjoint set.
- **Simple Extension** A (Σ, X)-simple extension is a timed word ρ over $2^X \cup \Sigma$ such that at any point $i \in dom(\rho)$, $\sigma_i \cap \Sigma \neq \emptyset$
- **Simple Projection** A timed word ρ over Σ obtained by deleting symbols in X from (Σ, X) extension ρ' is called its Simple Projection.



A valid Σ, X Extension $\Sigma = \{a, b, c\}$ $X = \{d\}$



The simple projection of the above Extension over Σ

Definitions: Oversampled Projection

Definitions: Oversampled Projection

- Let Σ, X be finite disjoint set.

Definitions: Oversampled Projection

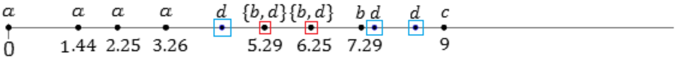
- Let Σ, X be finite disjoint set.
- **Oversampled Behaviour** A (Σ, X)-oversampled behaviour is a timed word over $2^X \cup \Sigma$, such that $\sigma'_1 \cap \Sigma \neq \emptyset$ and $\sigma'_{|dom(\rho')|} \cap \Sigma \neq \emptyset$.

Definitions: Oversampled Projection

- Let Σ, X be finite disjoint set.
- **Oversampled Behaviour** A (Σ, X)-oversampled behaviour is a timed word over $2^X \cup \Sigma$, such that $\sigma'_1 \cap \Sigma \neq \emptyset$ and $\sigma'_{|dom(\rho')|} \cap \Sigma \neq \emptyset$.
- **Oversampled Projection** A timed word ρ over Σ obtained by deleting symbols in X (and thus deleting the points containing only X)from (Σ, X) oversampled behaviour ρ' is called its Oversampled Projection.

Definitions: Oversampled Projection

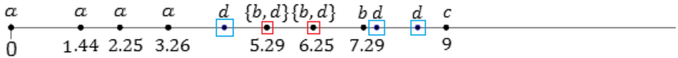
- Let Σ, X be finite disjoint set.
- **Oversampled Behaviour** A (Σ, X)-oversampled behaviour is a timed word over $2^X \cup \Sigma$, such that $\sigma'_1 \cap \Sigma \neq \emptyset$ and $\sigma'_{|dom(\rho')|} \cap \Sigma \neq \emptyset$.
- **Oversampled Projection** A timed word ρ over Σ obtained by deleting symbols in X (and thus deleting the points containing only X)from (Σ, X) oversampled behaviour ρ' is called its Oversampled Projection.



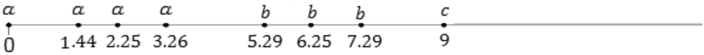
A valid Σ, X oversampled behaviour $\Sigma = \{a, b, c\}$ $X = \{d\}$

Definitions: Oversampled Projection

- Let Σ, X be finite disjoint set.
- **Oversampled Behaviour** A (Σ, X)-oversampled behaviour is a timed word over $2^X \cup \Sigma$, such that $\sigma'_1 \cap \Sigma \neq \emptyset$ and $\sigma'_{|dom(\rho')|} \cap \Sigma \neq \emptyset$.
- **Oversampled Projection** A timed word ρ over Σ obtained by deleting symbols in X (and thus deleting the points containing only X)from (Σ, X) oversampled behaviour ρ' is called its Oversampled Projection.



A valid Σ, X oversampled behaviour $\Sigma = \{a, b, c\}$ $X = \{d\}$



The oversampled projection of the above oversampled behaviour over Σ

We say that φ over Σ is equisatisfiable modulo temporal projection ψ over $\Sigma \cup 2^X$ iff:

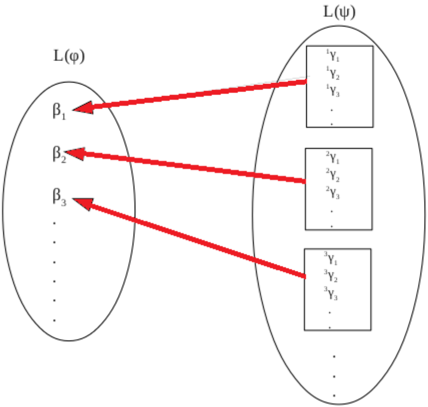


Figure: Figure Illustrating φ is equisatisfiable to ψ. Arrow represents the temporal(simple or oversampled) projection function

Flattening: An example

Flattening: An example

- Flattening is a technique to reduce the modal depth of the formula preserving satisfiability.

Flattening: An example

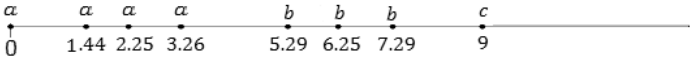
- Flattening is a technique to reduce the modal depth of the formula preserving satisfiability.
- Example Let $\phi = a \, \mathsf{U}_{[2,5]}(b \, \mathsf{U}_{[2,3]} c)$

Flattening: An example

- Flattening is a technique to reduce the modal depth of the formula preserving satisfiability.
- Example Let $\phi = a\,\mathsf{U}_{[2,5]}(b\,\mathsf{U}_{[2,3]}c)$
- $\phi_{flat} = a\,\mathsf{U}_{[2,5]}d \wedge \Box(d \leftrightarrow (b\,\mathsf{U}_{[2,3]}c)) \wedge \Box(d \rightarrow a \vee b \vee c)$

Flattening: An example

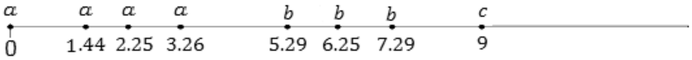
- Flattening is a technique to reduce the modal depth of the formula preserving satisfiability.
- Example Let $\phi = a \, \mathsf{U}_{[2,5]}(b \, \mathsf{U}_{[2,3]} c)$
- $\phi_{flat} = a \, \mathsf{U}_{[2,5]} d \wedge \Box(d \leftrightarrow (b \, \mathsf{U}_{[2,3]} c)) \wedge \Box(d \rightarrow a \vee b \vee c)$



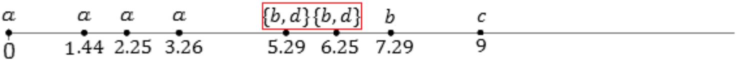
A Timed Word models φ

Flattening: An example

- Flattening is a technique to reduce the modal depth of the formula preserving satisfiability.
- Example Let $\phi = a\,\mathsf{U}_{[2,5]}(b\,\mathsf{U}_{[2,3]}c)$
- $\phi_{flat} = a\,\mathsf{U}_{[2,5]}d \wedge \Box(d \leftrightarrow (b\,\mathsf{U}_{[2,3]}c)) \wedge \Box(d \rightarrow a \vee b \vee c)$



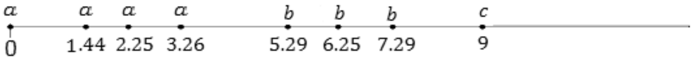
A Timed Word models φ



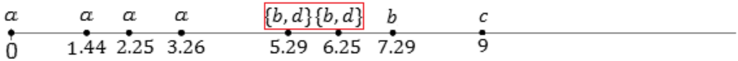
The corresponding Timed Word which is the model of flattened formula φ_{flat}

Flattening: An example

- Flattening is a technique to reduce the modal depth of the formula preserving satisfiability.
- Example Let $\phi = a\,\mathsf{U}_{[2,5]}(b\,\mathsf{U}_{[2,3]}c)$
- $\phi_{flat} = a\,\mathsf{U}_{[2,5]}d \wedge \Box(d \leftrightarrow (b\,\mathsf{U}_{[2,3]}c)) \wedge \Box(d \rightarrow a \vee b \vee c)$



A Timed Word models φ



The corresponding Timed Word which is the model of flattened formula φ_{flat}

Thus flattening is an example of a reduction preserving satisfiability modulo simple projections.

Related Work

Related Work

- Satisfiability Checking of *MITL* is decidable with *EXPSPACE* complexity. [Alur *et al. J.ACM* 1996]

Related Work

- Satisfiability Checking of *MITL* is decidable with *EXPSPACE* complexity. [Alur *et al*. *J.ACM* 1996]
- Satisfiability problem for MTL[$\mathsf{U_I}$] is decidable by reducing it to reachability problem for 1-clock Timed Alternating Automata.[Ouaknine *et al*. *LICS* 2005]

Related Work

- Satisfiability Checking of *MITL* is decidable with *EXPSPACE* complexity. [Alur *et al*. *J.ACM* 1996]
- Satisfiability problem for MTL[U_I] is decidable by reducing it to reachability problem for 1-clock Timed Alternating Automata.[Ouaknine *et al*. *LICS* 2005]
- Satisfiability Checking of QTL with counting is decidable with *EXPSPACE* complexity. [Rabinovich *et.al*. *FORMATS* 2008]

Related Work

- Satisfiability Checking of *MITL* is decidable with *EXPSPACE* complexity. [Alur *et al*. *J.ACM* 1996]
- Satisfiability problem for MTL[U_I] is decidable by reducing it to reachability problem for 1-clock Timed Alternating Automata.[Ouaknine *et al*. *LICS* 2005]
- Satisfiability Checking of QTL with counting is decidable with *EXPSPACE* complexity. [Rabinovich *et.al*. *FORMATS* 2008]
- Counting adds expressiveness to MITL over signals [Rabinovich *FORMATS* 2008].

Related Work

- Satisfiability Checking of *MITL* is decidable with *EXPSPACE* complexity. [Alur *et al*. *J.ACM* 1996]
- Satisfiability problem for MTL[$\mathrm{U_I}$] is decidable by reducing it to reachability problem for 1-clock Timed Alternating Automata.[Ouaknine *et al*. *LICS* 2005]
- Satisfiability Checking of QTL with counting is decidable with *EXPSPACE* complexity. [Rabinovich *et.al*. *FORMATS* 2008]
- Counting adds expressiveness to MITL over signals [Rabinovich *FORMATS* 2008].
- *MTL* with counting over signals is expressively complete with $FO[<, +1]$ over reals [Hunter *CSL* 2013].

Related Work

- Satisfiability Checking of *MITL* is decidable with *EXPSPACE* complexity. [Alur *et al*. *J.ACM* 1996]
- Satisfiability problem for MTL[U_I] is decidable by reducing it to reachability problem for 1-clock Timed Alternating Automata.[Ouaknine *et al*. *LICS* 2005]
- Satisfiability Checking of QTL with counting is decidable with *EXPSPACE* complexity. [Rabinovich *et.al*. *FORMATS* 2008]
- Counting adds expressiveness to MITL over signals [Rabinovich *FORMATS* 2008].
- *MTL* with counting over signals is expressively complete with $FO[<, +1]$ over reals [Hunter *CSL* 2013].
- Counting LTL is equivalent to LTL and has $EXP-SPACE$ complete satisfiability checking.[Laroussinie *et. al*. *TIME* 2010].

Our Results

Our Results

- Satisfiability Checking for CTMTL is decidable.

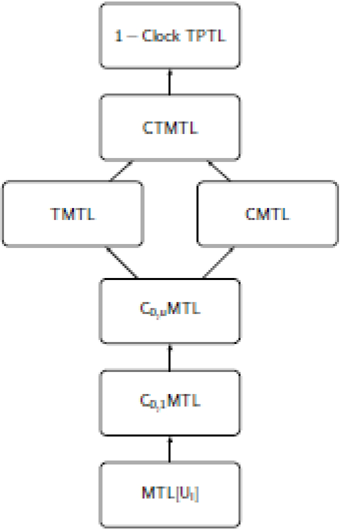
Our Results

- Satisfiability Checking for CTMTL is decidable.
- Exploring Expressiveness relations amongst fragments of MTL with counting over timed words(Pointwise Semantics).

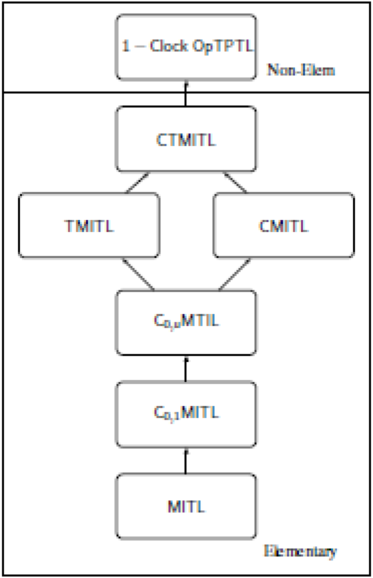
Presentation Flow

- Model : Timed Words
- Timed Logic with Counting : Syntax and Semantics
- Temporal Projections : Simple and Oversampled
- Expressiveness Relations with Counting Extensions
- Satisfiability Checking: Decidability
- Discussion
- Future Work

Expressiveness Heirarchy : Logic with counting

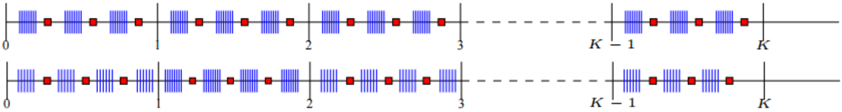


Expressiveness Heirarchy : Non-Punctual Fragments



TMTL − CMTL $\neq \emptyset$

$$\varphi = \Diamond_{(0,1),\#a \geq 3} b$$



Presentation Flow

- Model : Timed Words
- Timed Logic with Counting : Syntax and Semantics
- Temporal Projections : Simple and Oversampled
- Expressiveness Relations with Counting Extensions
- Satisfiability Checking: Decidability
- Discussion
- Future Work

Satisfiability Checking : Decidability

- **Flatten the formula**

- **Flatten the formula**
- All the counting modalities are of the form $\Box(w \leftrightarrow \mathsf{C}_I^{\sim n} a)$ and $\Box(w \leftrightarrow a \mathsf{U}_{I,\#x \sim n} b)$

- **Flatten the formula**
- All the counting modalities are of the form $\Box(w \leftrightarrow \mathsf{C}_I^{\sim n} a)$ and $\Box(w \leftrightarrow a\mathsf{U}_{I,\#x\sim n} b)$
- Next we eliminate counting modalities from the above flattened formula preserving satisfiability to show decidability.

Eliminating $C_I^{\geq n}b$ modality

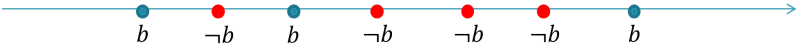
Eliminating $C_I^{\geq n} b$ modality

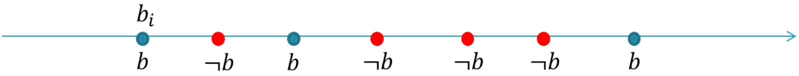
- Given a word ρ over Σ we construct a simple extension ρ' over $\Sigma \cup \{b_0, b_1, \ldots, b_{n-1}\}$

Eliminating $\mathsf{C}_I^{\geq n}\mathsf{b}$ modality

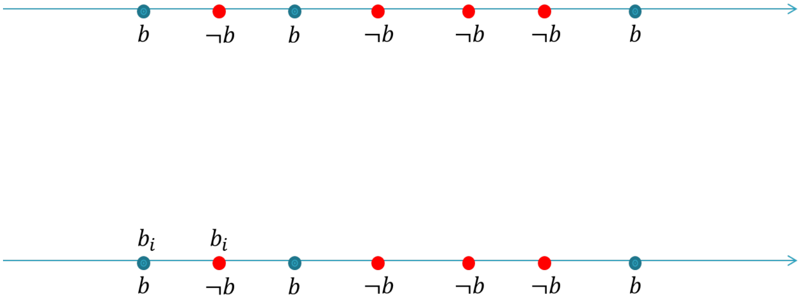
- Given a word ρ over Σ we construct a simple extension ρ' over $\Sigma \cup \{b_0, b_1, \ldots, b_{n-1}\}$
- $\{b_0, b_1, \ldots, b_{n-1}\}$ works as a counter. Using their behaviour we precisely mark a as the witness for $\mathsf{C}_I^{\geq n}\mathsf{b}$.

Construction of ρ'

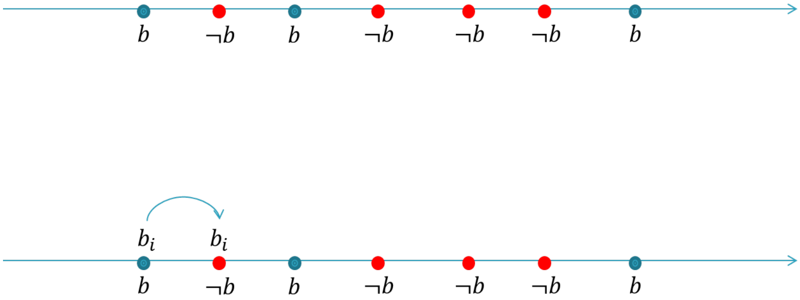




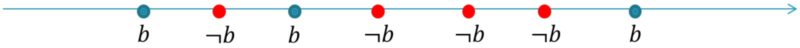
Construction of ρ'

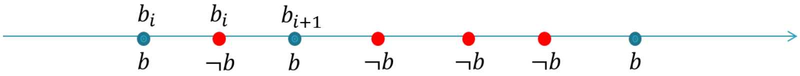


Construction of ρ'

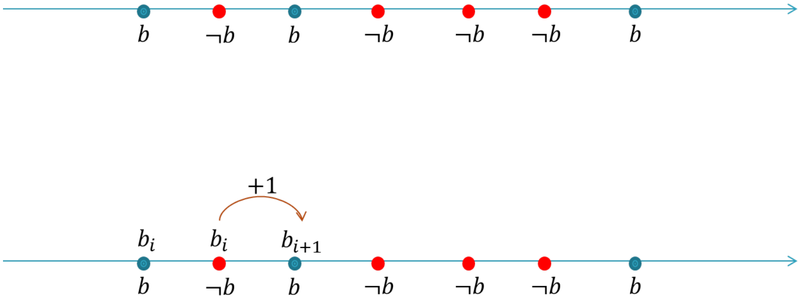


Construction of ρ'

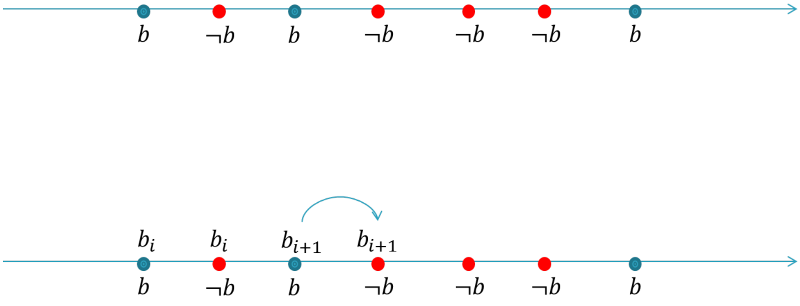




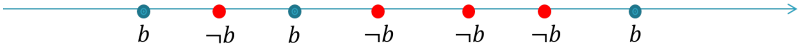
Construction of ρ'

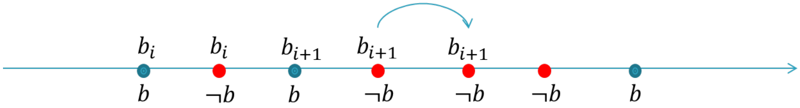


Construction of ρ'

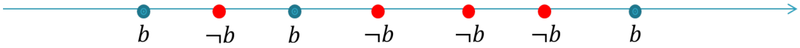


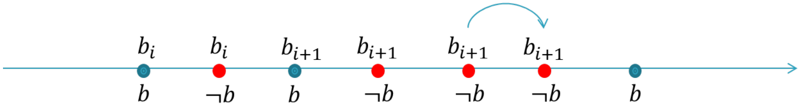
Construction of ρ'



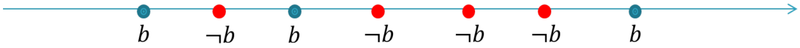


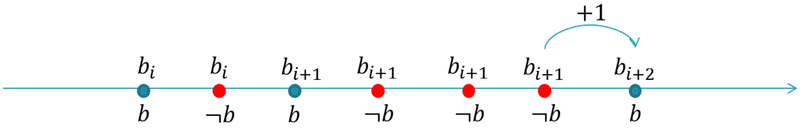
Construction of ρ'





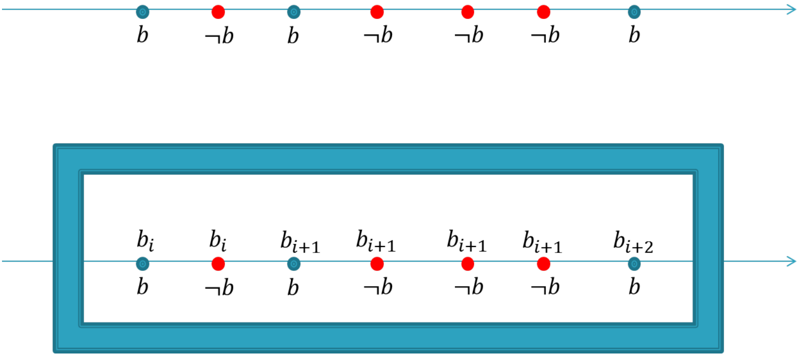
Construction of ρ'



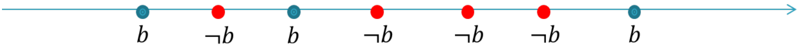


Construction of ρ'

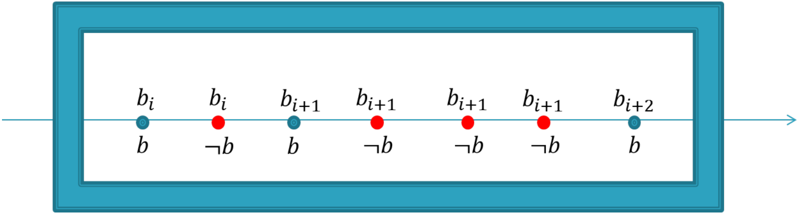
Note how the behaviour of b_i helps in finding the occurrences of b



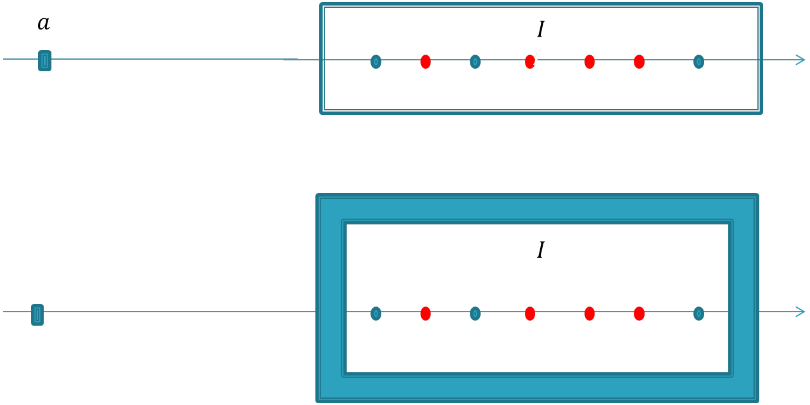
Construction of ρ'



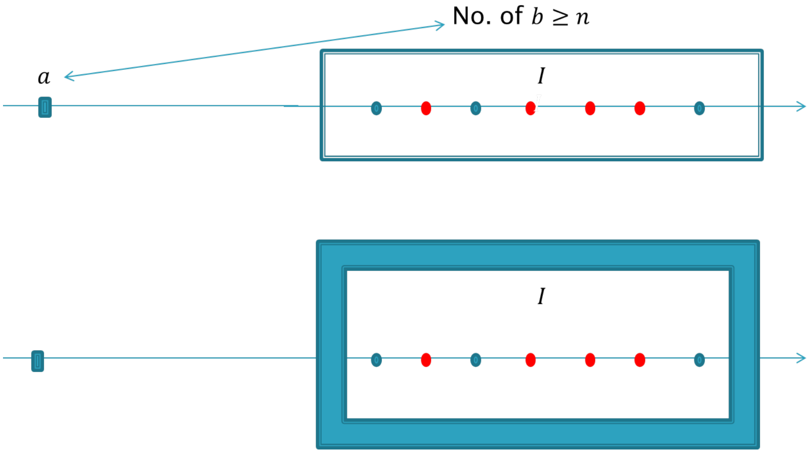
$No. of\ points\ where\ b\ holds\ = No. of\ different\ indices\ along\ with\ b$



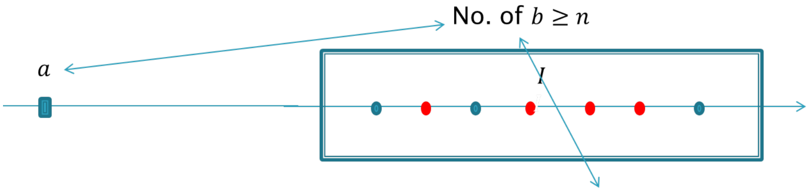
Construction of ρ'



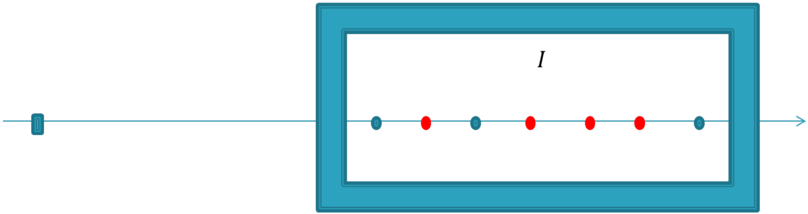
Construction of ρ'



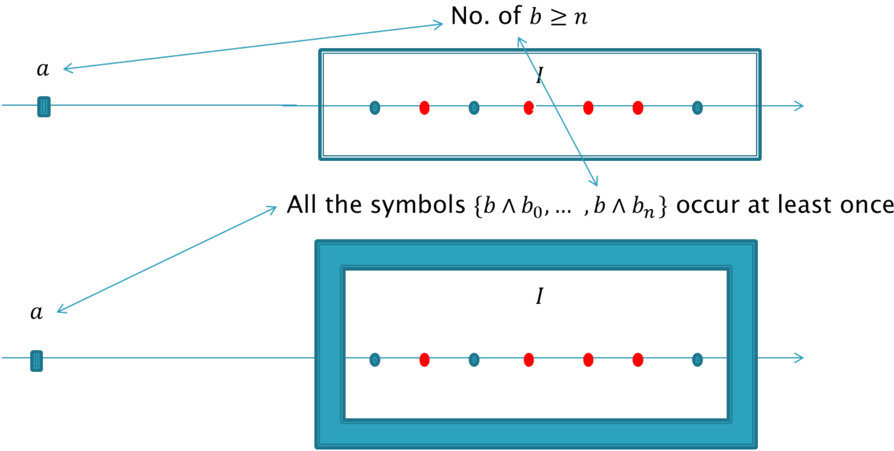
Construction of ρ'



All the symbols $\{b \wedge b_0, \ldots, b \wedge b_n\}$ occur at least once



Construction of ρ'



Eliminating UT modality

Eliminating UT modality

- Given a word ρ over Σ we construct a **oversampling** ρ' over $\Sigma \cup C \cup B$

Eliminating UT modality

- Given a word ρ over Σ we construct a **oversampling** ρ' over $\Sigma \cup C \cup B$
  - $C = \{c_0, \ldots, c_u\}$:

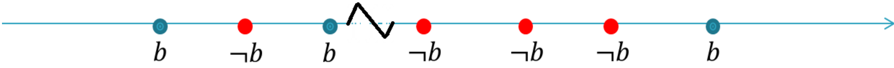
Eliminating UT modality

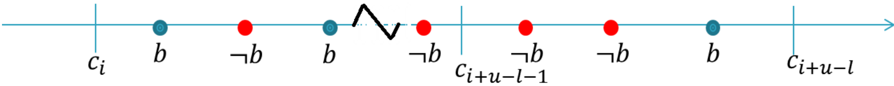
- Given a word ρ over Σ we construct a **oversampling** ρ' over $\Sigma \cup C \cup B$
  - $C = \{c_0, \ldots, c_u\}$:These propositions oversample the model at integer time stamps.

Eliminating UT modality

- Given a word ρ over Σ we construct a **oversampling** ρ' over $\Sigma \cup C \cup B$
  - $C = \{c_0, \ldots, c_u\}$:These propositions oversample the model at integer time stamps.
  - $B = \bigcup_{i=0}^{u} B^i$ where $B^i = \{b_0^i, \ldots b_n^i\}$: These propositions are used as counters for b. Counter B^i resets at integer point marked c_i and saturates once the value reaches n till the next reset.

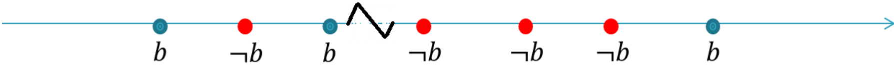
Oversample the word

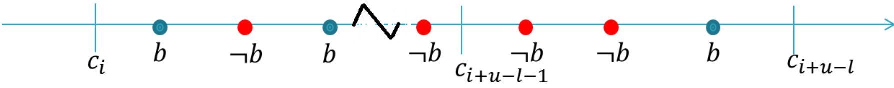




Construction of ρ'

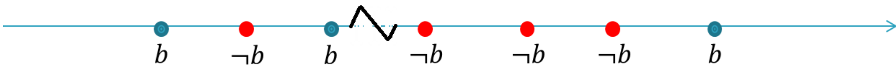
Oversample the word

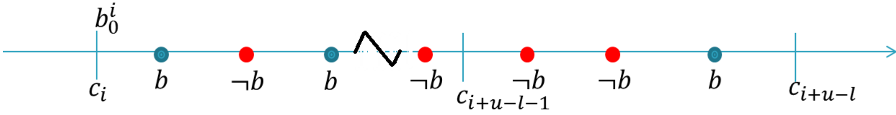




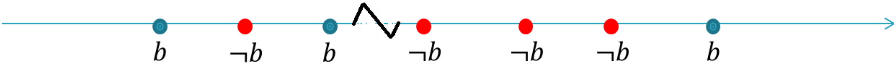
Construction of ρ'

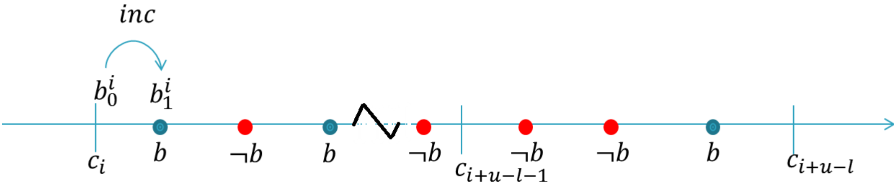
Initiate the counter





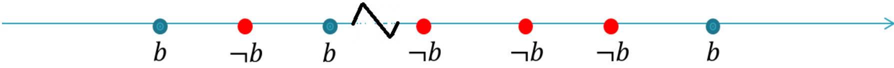
Propagate the counter

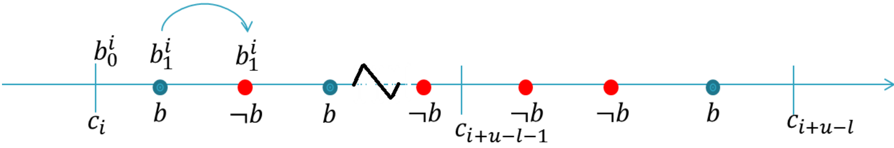




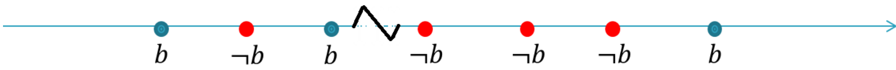
Construction of ρ'

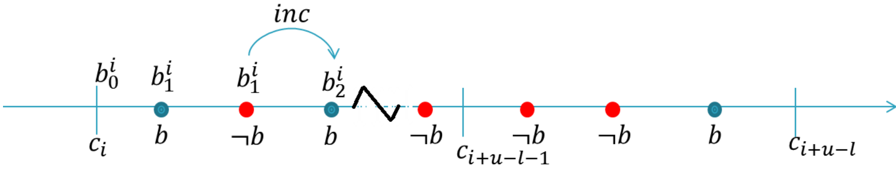
Propagate the counter





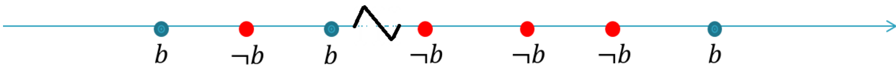
Propagate the counter

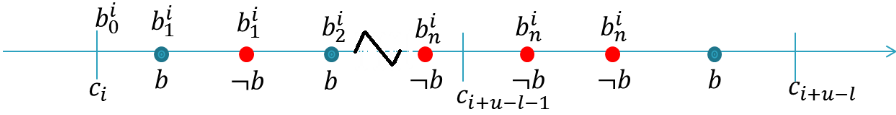




Construction of ρ'

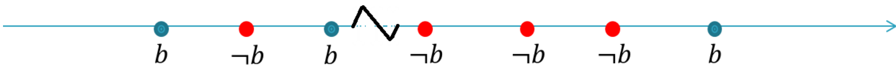
Propagate the counter

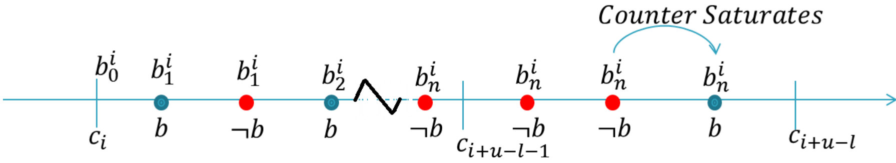




Construction of ρ'

Propagate the counter and stop incrementing after highest val

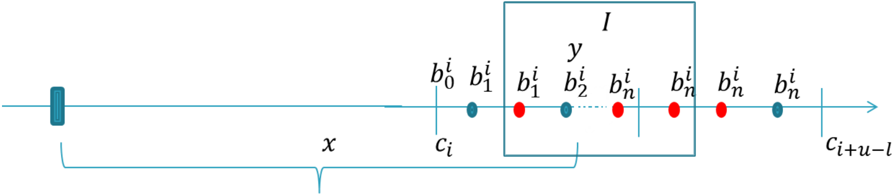




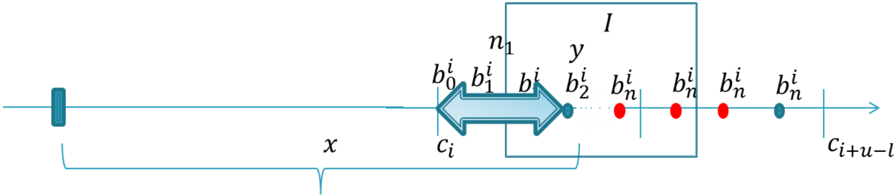
Marking Witness for UT sub-formula

Marking Witness for UT sub-formula

We count the number of occurrence of b in two stages:

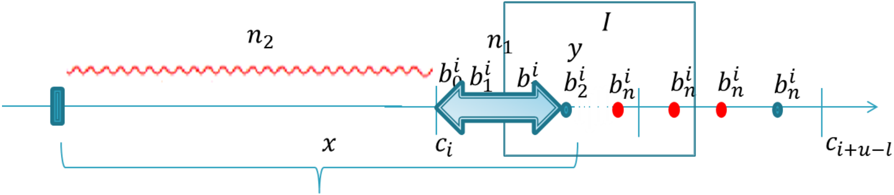


We count the number of occurrence of b in two stages:

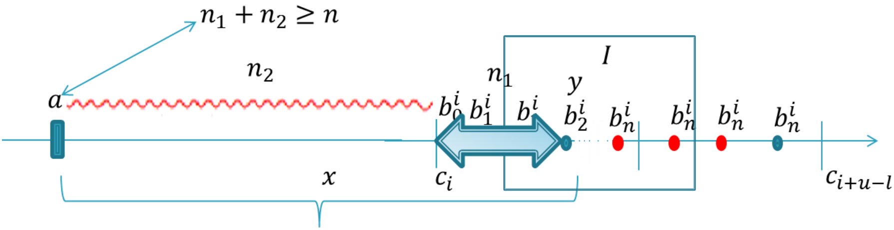


Marking Witness for UT sub-formula

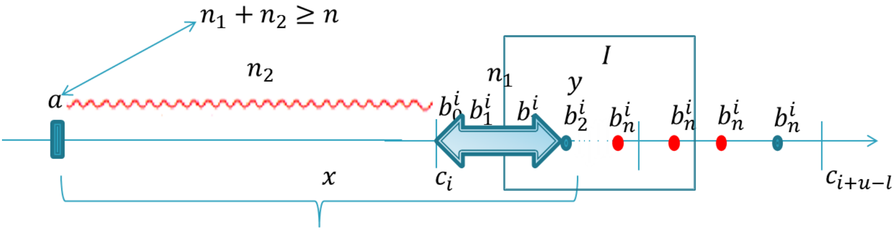
We count the number of occurrence of b in two stages:



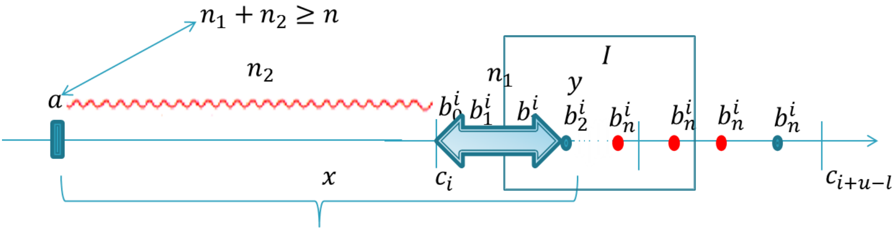
Marking Witness for UT sub-formula



Marking Witness for UT sub-formula



$n_1 \geq n \vee n_2 \geq n$



$n_1 \geq n \vee n_2 \geq n$ Or, $n_1 < n \wedge n_2 < n$ and thus bounded number of cases (disjunctions).

Presentation Flow

- Model : Timed Words
- Timed Logic with Counting : Syntax and Semantics
- Temporal Projections : Simple and Oversampled
- Expressiveness Relations with Counting Extensions
- Satisfiability Checking: Decidability
- Conclusion
- Future Work

Conclusion

Conclusion

- Two ways of extending MTL with counting threshold constraints is studied.

Conclusion

- Two ways of extending MTL with counting threshold constraints is studied.
- Both ways add expressiveness to MTL orthogonally.

Conclusion

- Two ways of extending MTL with counting threshold constraints is studied.
- Both ways add expressiveness to MTL orthogonally.
- Satisfiability checking for the logic CTMTL is decidable.

Conclusion

- Two ways of extending MTL with counting threshold constraints is studied.
- Both ways add expressiveness to MTL orthogonally.
- Satisfiability checking for the logic CTMTL is decidable.
- Both the extensions enjoy benefits of relaxing punctuality.

Conclusion

- Two ways of extending MTL with counting threshold constraints is studied.
- Both ways add expressiveness to MTL orthogonally.
- Satisfiability checking for the logic CTMTL is decidable.
- Both the extensions enjoy benefits of relaxing punctuality.
- Unlike continuous semantics, pointwise semantics creates a zoo of sub-logics in the expressiveness hierarchy.

Presentation Flow

- Model : Timed Words
- Timed Logic with Counting : Syntax and Semantics
- Temporal Projections : Simple and Oversampled
- Expressiveness Relations with Counting Extensions
- Satisfiability Checking: Decidability
- Conclusion
- Future Work

Future Work

- Exploring complexity results for satisfiability checking of CTMTL.
- Extending logics with modulo counting and study the expressiveness and satisfiability checking for those extensions.
- Complete picture of expressiveness of these counting extensions with different versions of past operators.
- Study model checking and synthesis problems for these extensions.

Thank You

References

- Y. Hirshfeld and A. Rabinovich. An expressive temporal logic for real time. In MFCS, pages 492–504, 2006.
- P. Hunter. When is metric temporal logic expressively complete? In CSL, pages 380–394, 2013.
- S. N. Krishna K. Madnani and P. K. Pandya. Partially punctual metric temporal logic is decidable. In TIME, pages 174–183, 2014.
- D. Kini, S. N. Krishna, and P. K. Pandya. On construction of safety signal automata for MITL[U, S] using temporal projections. In FORMATS, pages 225–239, 2011.
- F. Laroussinie, A. Meyer, and E. Petonnet. Counting ltl. In TIME, pages 51–58, 2010.
- K. Madnani, S. N. Krishna, and P. K. Pandya. Partially punctual metric temporal logic is decidable. In http://arxiv.org/abs/1404.6965, 2014.
- J. Ouaknine and J. Worrell. On the decidability of metric temporal logic. In LICS, pages 188–197, 2005.

References

- F. Chevalier P. Bouyer and N. Markey. On the expressiveness of tptl and mtl. In FST&TCS, pages 432–443, 2005.
- P. K. Pandya and S. Shah. On expressive powers of timed logics: Comparing boundedness, non-punctuality, and deterministic freezing. In CONCUR, pages 60–75, 2011.
- Pavithra Prabhakar and Deepak D'Souza. On the expressiveness of MTL with past operators. In FORMATS, pages 322–336, 2006.
- A. Rabinovich. Complexity of metric temporal logics with counting and the pnueli modalities. Theor. Comput. Sci., 411(22-24):2331–2342, 2010.
- Jean Francois Raskin. Logics, Automata and Classical Theories for Deciding Real Time. PhD thesis, Universite de Namur, 1999.