Expressiveness of Temporal Logics

François Laroussinie and Nicolas Markey

Lab. Specification et Verification
ENS Cachan & CNRS, France

August 4, 2006

Outline

1 Timed temporal logics
   • Definitions
   • Expressiveness and complexity

2 TPTL vs MTL

3 Timed logics and timed automata

Outline of today’s lecture

1 Timed temporal logics
   • Definitions
   • Expressiveness and complexity

2 TPTL vs MTL

3 Timed logics and timed automata

Timed temporal logics

Temporal logics = qualitative requirements

Timed temporal logics adds quantitative requirements.
Timed temporal logics

Temporal logics = qualitative requirements

**Timed** temporal logics adds quantitative requirements.

**Example**

Any request is granted **in at most 1 minute**.

An alarm rings if the doors are open **for more than 30 seconds**.

Requires explicit timing constraints in the model.

Adding “time” in Kripke structures

- basic idea: counting the number of transitions:

Examples

Adding “time” in Kripke structures

- basic idea: counting the number of transitions:

Examples
Adding “time” in Kripke structures

- basic idea: counting the number of transitions:
- slightly more involved: adding timing informations in Kripke structures:

Examples

```
G(go 3rd floor ⇒ F≤14 open3)
```

```
A G(E F≤25 open1)
```

⇝ those models are not very expressive (only more succinct);

⇝ in this settings, the logics also are not more expressive:

```
A G(E F≤25 open1) ≡ A G(E X(open1 ∨ E X(open1 ∨ E X(open1 ∨ E X(open1...)))))
```
Adding "time" in Kripke structures

- basic idea: counting the number of transitions:
- slightly more involved: adding timing informations in Kripke structures:

$$A G(\mathcal{F}_{\leq} \mathcal{G} \mathcal{E}(\text{open}_1)) \equiv A G(\mathcal{E}(\mathcal{X}(\text{open}_1 \lor \mathcal{E}(\mathcal{X}(\text{open}_1 \lor 
\mathcal{E}(\mathcal{X}(\text{open}_1 \lor 
\mathcal{E}(\mathcal{X}(\text{open}_1 \lor ...))))))))$$

we need a real-time extension of those models.

Timed automata

**Definition**

A *timed automaton* is a tuple $\mathcal{A} = (Q, Q_0, C, \to, \Sigma, \ell)$ s.t.:
- $Q$ is the set of locations, of which $Q_0$ are initial;
- $C$ is a (finite) set of *clock variables*;
- $\to$ is the set of transitions
- $\Sigma$ is the alphabet;
- $\ell$ labels either the states or the transitions.

Clocks are used on transitions: a transition is labeled with a *guard*, i.e., a list of constraints $x \sim n$ where $x \in C$, $n \in \mathbb{Z}^+$ and $\sim \in \{<, \leq, =, \geq, >\}$.

Timed automata

**Example**

A timed automaton is a tuple $\mathcal{A} = (Q, Q_0, C, \to, \Sigma, \ell)$ s.t.:
- $Q$ is the set of locations, of which $Q_0$ are initial;
- $C$ is a (finite) set of *clock variables*;
- $\to$ is the set of transitions
- $\Sigma$ is the alphabet;
- $\ell$ labels either the states or the transitions.
A timed word is a function $w : \mathbb{Z}^+ \to 2^{AP} \times \mathbb{R}^+$ such that $w_2$ is nondecreasing and diverges.

Example

$w = (\varepsilon, 0) (\text{green}, 0.6) (\text{blue}, 0.8) (\text{red}, 1.1) (\text{blue}, 1.6) (\text{green}, 1.6) \ldots$
Timed automata

Example

-q
\[ z \leq 1 \]
\[ y := 0 \]
\[ y < 1, x := 0 \]
\[ y = 1, z := 0 \]
\[ y = 1, y := 0 \]

1 2
\[ x = \]
\[ y = \]
\[ z = \]
0
0
0
0.6
0
0.6
0.8
0.2
0.8
0.5
0
0
0
Timed automata

Example

\begin{align*}
  x &= 0 & 0.6 & 0.8 & 0 \\
  y &= 0 & 0 & 0.2 & 0.5 \\
  z &= 0 & 0.6 & 0.8 & 1.1
\end{align*}

Timed automata

Example

\begin{align*}
  x &= 0 & 0.6 & 0.8 & 1.1 \\
  y &= 0 & 0 & 0.2 & 0.5 \\
  z &= 0 & 0.6 & 0.8 & 1.1
\end{align*}

Timed automata

Example

\begin{align*}
  x &= 0 & 0.6 & 0.8 & 0 & 0.5 \\
  y &= 0 & 0 & 0.2 & 0.5 & 1 \\
  z &= 0 & 0.6 & 0.8 & 1.1 & 1.6
\end{align*}

Timed automata

Example

\begin{align*}
  x &= 0 & 0.6 & 0.8 & 0 & 0.5 \\
  y &= 0 & 0 & 0.2 & 0.5 & 1 \\
  z &= 0 & 0.6 & 0.8 & 1.1 & 0
\end{align*}
Timed automata

Example

Definition
A timed state sequence is a function $\pi : \mathbb{R}^+ \to 2^{AP}$.

Example

Timed automata can also be used for defining languages: it suffices to add an acceptance (e.g., Büchi) condition.

Example

This automaton accepts timed words in which there exists an occurrence of green that is not followed, one time unit later, by another occurrence of green:

$$F(\text{green} \land \neg F_{=1} \text{green}).$$
Timed automata

The negation of

\[ F(\text{green} \land \neg F_{=1} \text{green}) \]

is

\[ G(\text{green} \Rightarrow F_{=1} \text{green}) \]

Lemma

No Büchi timed automaton accepts the language defined by

\[ G(\text{green} \Rightarrow F_{=1} \text{green}) \]

Theorem

Timed automata are closed under conjunction and disjunction, but not under negation.

Extending temporal logics with time

Two different ways of extending temporal logics:

- by associating intervals with modalities: those intervals (having rational bounds) indicate e.g. the moment at which an eventuality is to be fulfilled.

Examples

\[ G(\text{call}_3 \Rightarrow F_{[0,1]} \text{open}_3) \]

\[ A G(E F_{[0,1]} \text{open}_1) \]
Two different ways of extending temporal logics:
- **by associating intervals with modalities**: those intervals (having rational bounds) indicate e.g. the moment at which an eventuality is to be fulfilled.
- **by using real clocks in the formula**: the clocks can be reset at some point during the evaluation of the formula, and then compared to rationals.

### Syntax of MTL

\[ \text{MTL} \ni \varphi ::= p \mid \neg \varphi \mid \varphi \lor \varphi \mid \varphi \U \varphi \]

where \( p \) ranges over \( \mathcal{AP} \) and \( I \) is an interval with bounds in \( \mathbb{Q}^+ \cup \{+\infty\} \).

### Pointwise semantics of MTL

over \( \pi = \{(w_i), (t_i)\} \):

- \( \pi, i \models \varphi \Leftrightarrow \) there exists some \( j > 0 \) s.t.
  - \( \pi, j + k \models \psi \) for all \( 0 < k < j \).
  - \( t_{i,j} - t_i \in I \).

### Examples

\[ G(\text{call}_3 \Rightarrow x.F(\text{open}_3 \land x \leq 1)) \]
\[ A G(x.E F(\text{open}_1 \land x \leq 3)) \]
Timed logics in the pointwise framework

Syntax of MTL:

\[ \phi \in \{ p | \neg \phi | \phi \lor \psi | \phi U_j \phi \} \]

where \( p \) ranges over \( AP \) and \( J \) is an interval with bounds in \( Q^+ \cup \{-\infty\} \).

Pointwise semantics of MTL: over \( \pi = ((w_j), (t_j)) \):
- \( \pi, i \models \phi \lor \psi \) iff there exists some \( j > 0 \) s.t.
- \( \pi, j + k \models \psi \) for all \( 0 < k < j \).
- \( t_{ij} - t_i \in I \).

Examples:

\[
\begin{array}{c}
0 & 1 & 2 \\
\text{F} & \text{G} & \text{F} \\
\text{green} & \text{blue} & \text{green}
\end{array}
\]

Timed logics in the pointwise framework

Syntax of TPTL:

\[ \phi \in \{ p | \neg \phi | \phi \lor \psi | \phi U_j \phi \} \]

where \( p \) ranges over \( AP \) and \( J \) is an interval with bounds in \( Q^+ \cup \{-\infty\} \).

Pointwise semantics of MTL: over \( \pi = ((w_j), (t_j)) \):
- \( \pi, i \models \phi \lor \psi \) iff there exists some \( j > 0 \) s.t.
- \( \pi, j + k \models \psi \) for all \( 0 < k < j \).
- \( t_{ij} - t_i \in I \).

Examples:

\[
\begin{array}{c}
0 & 1 & 2 \\
\text{F} & \text{G} & \text{F} \\
\text{red} & \text{blue} & \text{red}
\end{array}
\]
Timed logics in the pointwise framework

Syntax of TPTL:

\[ \text{TPTL} \models \varphi ::= p | x \sim c | \neg \varphi | \varphi \lor \varphi | \varphi \mathbf{U} \varphi | x. \varphi \]

where \( p \) ranges over \( AP \), \( x \) ranges over a set of formula clocks, \( c \in \mathbb{Q}_+^\ast \) and \( \sim \in \{<, \leq, =, \geq, >\} \).

Pointwise semantics of TPTL: over \( \pi = ((w_i), (t_i)) \):

\[ \pi, i, \tau \models x \sim c \iff \tau(x) \sim c \]

\[ \pi, i, \tau \models x \models \varphi \iff \tau(x) \models \varphi \]

\[ \pi, i, \tau \models \varphi \mathbf{U} \psi \iff \text{there exists some } j > 0 \text{ s.t.} \]

\[ \quad \pi, i + j, \tau + t_{i + j} = t_i \models \psi, \]

\[ \quad \pi, i + k, \tau + t_{i + k} = t_i \models \varphi \text{ for all } 0 < k < j. \]

Examples:

0 1 2
(red,0.3) (red,1.2) (blue,2.1)
\[ x.(\text{red U (blue } \land x \in [2, 3])) \]

Timed logics in the pointwise framework

Syntax of TPTL:

\[ \text{TPTL} \models \varphi ::= p | x \sim c | \neg \varphi | \varphi \lor \varphi | \varphi \mathbf{U} \varphi | x. \varphi \]

where \( p \) ranges over \( AP \), \( x \) ranges over a set of formula clocks, \( c \in \mathbb{Q}_+^\ast \) and \( \sim \in \{<, \leq, =, \geq, >\} \).

Pointwise semantics of TPTL: over \( \pi = ((w_i), (t_i)) \):

\[ \pi, i, \tau \models x \sim c \iff \tau(x) \sim c \]

\[ \pi, i, \tau \models x \models \varphi \iff \tau(x) \models \varphi \]

\[ \pi, i, \tau \models \varphi \mathbf{U} \psi \iff \text{there exists some } j > 0 \text{ s.t.} \]

\[ \quad \pi, i + j, \tau + t_{i + j} = t_i \models \psi, \]

\[ \quad \pi, i + k, \tau + t_{i + k} = t_i \models \varphi \text{ for all } 0 < k < j. \]

Examples:

0 1 2
(red,0.2) (green,1.1) (red,2.1)
\[ F(\text{green } \land x.(\bot \mathbf{U} (\text{red } \land x = 1))) \]
Timed logics in the continuous framework

Syntax of MTL:

\[ MTL \triangleright \varphi ::= p \mid \neg \varphi \mid \varphi \lor \varphi \mid \varphi U \varphi \]

Continuous semantics of MTL:

\[ \pi, t \models \varphi \quad \text{iff} \quad \exists u > 0 \text{ s.t.} \]
\[ - \pi, t + u \models \psi, \quad - \pi, t + v \models \varphi \text{ for all } 0 < v < u, \quad - u \in I. \]

Examples:

- \( F_{=2} \) green \( \equiv F_{=1}(F_{=1} \text{ green}) \)
Timed logics in the continuous framework

Syntax of TPTL:

\[
\text{TPTL} \ni \varphi ::= p \mid x \sim c \mid \neg \varphi \mid \varphi \lor \varphi \mid \varphi \mathbf{U} \varphi \mid x. \varphi
\]

Continuous semantics of TPTL: over \( \pi : \mathbb{R}^+ \rightarrow 2^\mathbb{AP} \):

- \( \pi, t, \tau \models x \sim c \) iff \( \tau(x) \sim c \)
- \( \pi, t, \tau \models x. \varphi \) iff \( \pi, i, \tau[x \leftarrow 0] \models \varphi \)
- \( \pi, t, \tau \models \varphi \mathbf{U} \psi \) iff there exists some \( u > 0 \) s.t.
  - \( \pi, t+u, \tau+u \models \psi \)
  - \( \pi, i + k, \tau + v - t \models \varphi \) for all \( 0 < v < u \).

Examples:

\[ x.((\text{red} \lor \text{blue}) \mathbf{U} (\text{green} \land x \leq 2)) \]
Outline

1 Timed temporal logics
   • Definitions
   • Expressiveness and complexity

2 TPTL vs MTL

3 Timed logics and timed automata

Outline of today’s lecture

1 Timed temporal logics
   • Definitions
   • Expressiveness and complexity

2 TPTL vs MTL

3 Timed logics and timed automata

MTL and TPTL are very expressive

Lemma
The halting problem for a Turing machine can be encoded in TPTL and MTL (with past) in both (pointwise and continuous) frameworks.

Proof (sketch).
- the successive configurations of the Turing machine are encoded on a one-time-unit-long segment;
- a transition of the Turing machine is applied between one configuration and its successor;
MTL and TPTL are very expressive

Lemma
The halting problem for a Turing machine can be encoded in TPTL and MTL (with past) in both (pointwise and continuous) frameworks.

Proof (sketch).

- the successive configurations of the Turing machine are encoded on a one-time-unit-long segment;
- a transition of the Turing machine is applied between one configuration and its successor;
- the final state of the Turing machine is eventually reached.

MTL and TPTL are very expressive

Lemma
The halting problem for a Turing machine can be encoded in TPTL and MTL (with past) in both (pointwise and continuous) frameworks.

Proof (sketch).

- the successive configurations of the Turing machine are encoded on a one-time-unit-long segment;
- a transition of the Turing machine is applied between one configuration and its successor;
- the final state of the Turing machine is eventually reached.

MTL and TPTL are very expressive

Theorem
Satisfiability of an MTL- or TPTL-formula is undecidable.

Definition
MITL is a (syntactic) fragment of MTL where punctuality is not allowed: intervals cannot be singletons.

Theorem
In the continuous semantics, with any MITL formula, we can associate a timed automaton that accepts exactly the same set of timed state sequences.
MTL and TPTL are very expressive

**Theorem**

Satisfiability of an MTL- or TPTL-formula is undecidable.

**Definition**

MITL is a (syntactic) fragment of MTL where punctuality is not allowed: intervals cannot be singletons.

**Theorem (Alur, Feder, Henzinger, 1991)**

In the continuous semantics, satisfiability of an MITL formula is EXPSPACE-complete.

**Corollary**

In the continuous semantics, there is no translation from MTL into MITL.

Outline

1. Timed temporal logics
   - Definitions
   - Expressiveness and complexity
2. TPTL vs MTL
3. Timed logics and timed automata

Relative expressiveness of TPTL and MTL

Clearly, MTL can be translated into TPTL:

\[ \varphi U_I \psi \equiv x. \varphi (\psi \land x \in I). \]

Conversely, consider the following TPTL formula:

\[ G[green \Rightarrow x. F(red \land F(blue \land x \leq 2))]. \]

It characterizes the following pattern:

\[ 0 \quad 1 \quad 2 \]

\[ \text{green} \quad \text{red} \quad \text{blue} \]
Conversely, consider the following TPTL formula:

\[ G[\text{green} \Rightarrow x. F(\text{red} \land F(\text{blue} \land x \leq 2))]. \]

It characterizes the following pattern:

\[
\begin{array}{ccc}
0 & 1 & 2 \\
\text{green} & \text{red} & \text{blue}
\end{array}
\]

**Conjecture (Alur, Henzinger, 1990)**

**Formula**

\[ G[\text{green} \Rightarrow x. F(\text{red} \land F(\text{blue} \land x \leq 2))]. \]

cannot be expressed in MTL.

Relative expressiveness of TPTL and MTL

In fact, the formula can be expressed:

\[
G[\text{green} \Rightarrow \begin{cases}
F[0,1] \text{red} \land F[1,2] \text{blue} \\
∨
F[0,1](\text{red} \land F[0,1] \text{blue}) \\
∨
F[1,2](\text{blue} \land F□1[□1,0] \text{red})
\end{cases}]
\]
Relative expressiveness of TPTL and MTL

In fact, the formula can be expressed:

\[ G(\text{green} \Rightarrow \begin{cases} & F_{[0,1]} \text{red} \land F_{[1,2]} \text{blue} \\ & F_{[0,1]}(\text{red} \land F_{[0,1]} \text{blue}) \end{cases}) \]

Lemma (Bouyer, Chevalier, Markey, 2005)
The formula can be expressed
- in MTL in the continuous framework,
- in MITL +Past in the pointwise framework.

The pointwise framework

Theorem (Bouyer, Chevalier, Markey, 2005)
TPTL is strictly more expressive than MTL in the pointwise semantics.

Proof (sketch).
- Let \( \psi = x \cdot F(\text{red} \land \text{blue} \land x \leq 2) \)
- Let \( \psi \) be an MTL formula.
  - \( n \) = temporal height of \( \psi \) = maximum number of nested modalities in \( \psi \)
  - \( p \) = granularity of \( \psi \) = inverse of the least common denominator of the constants appearing in \( \psi \).
- we assume that the constants of \( \psi \) are multiples of \( p \).
The pointwise framework

**Theorem (Bouyer, Chevalier, Markey, 2005)**

TPTL is strictly more expressive than MTL in the pointwise semantics.

**Proof (sketch).**

\[ A_p, n \models \phi \iff B_{p, n} \models \phi \]

**Lemma**

For any formula \( \psi \) of MTL with temporal height \( n \) and granularity \( p \),

\[ A_{n+3, p, 0} \models \psi \iff B_{n+3, p, 0} \models \psi \]

The continuous framework

**Theorem (Bouyer, Chevalier, Markey, 2005)**

TPTL is strictly more expressive than MTL in the continuous semantics.

**Proof (sketch).**

Let \( \varphi = x. F(\text{green} \land x \leq 1 \land G(x \leq 1 \Rightarrow \neg \text{blue})) \)

\[ A_p \models \varphi \]

**Corollary**

In the pointwise framework:

- MTL+Past is strictly more expressive than MTL.
- MITL+Past is strictly more expressive than MITL.
The continuous framework

**Theorem (Bouyer, Chevalier, Markey, 2005)**

TPTL is strictly more expressive than MTL in the continuous semantics.

**Proof (sketch).**

Let \( \varphi = x \cdot F(green \land x \leq 1 \land G(x \leq 1 \Rightarrow \neg blue)) \)

\[ A_{p,n} \quad B'_{p,n} \]

\[ n \text{ times green} \]

**Lemma**

The formula \( \varphi \) distinguishes between \( A_{p,n} \) and \( B'_{p,n} \).

**Corollary**

In the continuous framework:
- MTL+Past is strictly more expressive than MTL.
- MITL+Past is strictly more expressive than MITL.

---

Outline of today’s lecture

1. Timed temporal logics
   - Definitions
   - Expressiveness and complexity
   - TPTL vs MTL
   - Timed logics and timed automata

---

Outline

1. Timed temporal logics
   - Definitions
   - Expressiveness and complexity
   - TPTL vs MTL
   - Timed logics and timed automata
Two-way timed automata

In the sequel, we assume the pointwise semantics.

Definition (Alur & Henzinger, 1992)

A two-way timed automaton is a tuple $\mathcal{A} = (Q, Q_0, C, \rightarrow, \Sigma, \ell)$ where
- $Q, Q_0, C, \Sigma$ and $\ell$ are similar to the case of classical timed automata,
- each transition in $\rightarrow$ is decorated with a direction, telling whether the letter to be read at the next step is the one ahead or the one before.

Two-way timed automata might also have an acceptance condition.

Example

(q0, green, x:=0, y:=0) (q1, red, x<1) (q2, blue, y:=0) (q3, yellow, y>2)...

($\text{green}, 0.2$)($\text{blue}, 0.6$)($\text{red}, 1.1$)($\text{yellow}, 3.1$)...

Two-way timed automata

In the sequel, we assume the pointwise semantics.

Definition (Alur & Henzinger, 1992)

A two-way timed automaton is a tuple $\mathcal{A} = (Q, Q_0, C, \rightarrow, \Sigma, \ell)$ where
- $Q, Q_0, C, \Sigma$ and $\ell$ are similar to the case of classical timed automata,
- each transition in $\rightarrow$ is decorated with a direction, telling whether the letter to be read at the next step is the one ahead or the one before.

Two-way timed automata might also have an acceptance condition.

Example
Two-way timed automata

Example

Theorem (Alur & Henzinger, 1992)

Two-way timed automata are more expressive than one-way timed automata.

Proof (sketch).
The language accepted by the following MTL formula is known not to be recognizable by a timed automaton:

\[ G(\text{green} \Rightarrow F_{=1} \text{green}) \]

It is accepted by the following two-way timed automaton:

Bounded timed automata

Definition

A two-way timed automaton is k-bounded if each letter of the input word is read at most 2k + 1 times (for any input word).
### Bounded timed automata

**Definition**

A two-way timed automaton is **k-bounded** if each letter of the input word is read at most $2k + 1$ times (for any input word).

**Example**

This automaton is 1-bounded:

![Diagram of a 1-bounded two-way timed automaton]

### MITL+Past and k-bounded two-way timed automata

**Definition**

We write $\text{MITL}+\text{Past}_k$ for the fragment of MITL+Past where the number of alternations of $U$ and $S$ is at most $k - 1$.

**Example**

$$\text{green } U_{[1,2]} ((\text{green } S_{[3,\infty)}) \text{ red }) S_{[0,1]} (\text{red } U \text{ blue}))$$

**Theorem (Alur & Henzinger, 1992)**

With any formula in $\text{MITL}+\text{Past}_k$, we can associate a deterministic $k$-bounded two-way timed automaton that accepts exactly the same set of timed words.
**MITL+Past and $k$-bounded two-way timed automata**

**Definition**
We write $\text{MITL}+\text{Past}_k$ for the fragment of MITL+Past where the number of alternations of $U$ and $S$ is at most $k - 1$.

**Theorem (Alur & Henzinger, 1992)**
With any formula in MITL+Past$_k$, we can associate a deterministic $k$-bounded two-way timed automaton that accepts exactly the same set of timed words.

**Corollary**
The satisfiability problem for MITL+Past is PSPACE-complete.

Moreover, it can be proved that the hierarchy is strict:

**Theorem (Alur & Henzinger, 1992)**

\[
\text{MITL}+\text{Past}_k \subsetneq \text{MITL}+\text{Past}_{k+2}
\]