Weighted Timed Automata: Model-Checking and Games

Patricia Bouyer

LSV – CNRS & ENS de Cachan – France

Based on joint works with Thomas Brihaye, Ed Brinksma, Véronique Bruyère, Franck Cassez, Emmanuel Fleury, François Laroussinie, Kim G. Larsen, Nicolas Markey, Jean-François Raskin, and Jacob Illum Rasmussen
Outline

1. Introduction

2. Model-checking weighted timed automata

3. Optimal timed games

4. Conclusion
Model-checking

Does the system satisfy the property?

Modelling
Model-checking

Does the system satisfy the property?

Modelling

Model-checking Algorithm
Controller synthesis

Can we guide the system so that it satisfies the property?

Modelling
Controller synthesis

Can we guide the system so that it satisfies the property?

Modelling

Controller synthesis
Controller synthesis

Can we guide the system so that it satisfies the property?

Controller synthesis modeled as two player games
Timed automata

$x, y : \text{clocks}$

$x \leq 5, \ a, \ y := 0$

$y > 1, \ b, \ x := 0$
Model of weighted/priced timed automata

\[ P \xrightarrow{\ell} P' \]

\[ g, a, C := 0 \]

Cost rate

Discrete cost

[1] HSCC'01
Model of weighted/-priced timed automata

Cost rate

\[ P \]

Discrete cost

\[ p \]

\[ g, a, C := 0 \]

- A configuration: \((\ell, \nu)\)
- Two kinds of transitions:

\[
\begin{aligned}
(\ell, \nu) &\xrightarrow{\delta(d)} (\ell, \nu + d) \\
(\ell, \nu) &\xrightarrow{a} (\ell', \nu') \quad \text{where} \quad \\
v &\models g \\
v' &\equiv [C \leftarrow 0]v
\end{aligned}
\]

For some \(\ell \xrightarrow{g,a,C:=0} \ell'\)
Model of weighted/-priced timed automata

- a configuration: \((\ell, v)\)
- two kinds of transitions:

\[
\begin{align*}
\begin{cases}
(\ell, v) & \xrightarrow{\delta(d)} (\ell, v + d) \\
(\ell, v) & \xrightarrow{a} (\ell', v') 
\end{cases}
\end{align*}
\]

where

\[
\begin{align*}
\exists v \models g, v' &= [C \leftarrow 0]v \\
& \text{for some } \ell \xrightarrow{g, a, C := 0} \ell'
\end{align*}
\]

\[
\begin{align*}
\text{Cost} \left( (\ell, v) \xrightarrow{\delta(d)} (\ell, v + d) \right) &= P \cdot d \\
\text{Cost} \left( (\ell, v) \xrightarrow{a} (\ell', v') \right) &= p
\end{align*}
\]

\[
\text{Cost}(\rho) = \text{accumulated cost along run } \rho
\]
An example

[Larsen, Behrmann, Brinksma, Fehnker, Hune, Pettersson, Romijn – CAV’01]

**Fig. 2.** Figure (a) depicts the cost of landing a plane at time $t$. Figure (b) shows an LPTA modelling the landing costs. Figure (c) shows an LPTA model of the runway.
An example

\[\ell \leq 2; c; y := 0\]

\[\text{cost}(\ell_0) = 5\]

\[\ell_1\]

\[\text{cost}(\ell_1) = 1\]

\[\ell_2\]

\[x \geq 2; c; \text{cost} = 1\]

\[\ell_3\]

\[x \geq 2; c; \text{cost} = 7\]

\[\text{cost}(\ell_3) = 1\]
An example

\[ \ell \leq 2; c; y := 0 \]
\[ \text{cost}(\ell_0) = 5 \]
\[ \ell_1 \]
\[ y = 0 \]
\[ \text{cost}(\ell_1) = 1 \]
\[ \ell_2 \]
\[ x \geq 2; c; \text{cost} = 1 \]
\[ \text{cost}(\ell_2) = 10 \]
\[ \ell_3 \]
\[ x \geq 2; c; \text{cost} = 7 \]
\[ \text{cost}(\ell_3) = 1 \]

**Question:** what is the optimal cost for reaching the happy state?
An example

\[
\ell_0 \quad x \leq 2; \ c; \ y := 0 \\
\text{cost}(\ell_0) = 5
\]

\[
\ell_1 \quad y = 0
\]

\[
\ell_2 \quad x \geq 2; \ c; \ \text{cost} = 1 \\
\text{cost}(\ell_2) = 10
\]

\[
\ell_3 \quad x \geq 2; \ c; \ \text{cost} = 7 \\
\text{cost}(\ell_3) = 1
\]

**Question:** what is the optimal cost for reaching the happy state?

\[
5t + 10(2 - t) + 1
\]
An example

Question: what is the optimal cost for reaching the happy state?

\[ 5t + 10(2 - t) + 1, \ 5t + (2 - t) + 7 \]
An example

\[
\ell_0 \xrightarrow{x \leq 2; \ c; \ y := 0} \ell_1 \quad \text{cost}(\ell_0) = 5
\]

\[
\ell_1 \xrightarrow{y = 0} \ell_2 \quad \text{cost}(\ell_1) = \text{cost}(\ell_2) = 10
\]

\[
\ell_2 \xrightarrow{x \geq 2; \ c; \ \text{cost} = 1} \ell_3 \quad \text{cost}(\ell_3) = 1
\]

\[
\ell_3 \xrightarrow{x \geq 2; \ c; \ \text{cost} = 7} \text{happy state}
\]

**Question:** what is the optimal cost for reaching the happy state?

\[
\min \left( 5t + 10(2 - t) + 1 , \ 5t + (2 - t) + 7 \right)
\]
An example

Question: what is the optimal cost for reaching the happy state?

\[
\inf_{0 \leq t \leq 2} \min \left( 5t + 10(2 - t) + 1, 5t + (2 - t) + 7 \right) = 9
\]
An example

Question: what is the optimal cost for reaching the happy state?

\[ \inf_{0 \leq t \leq 2} \min \left( 5t + 10(2 - t) + 1, \ 5t + (2 - t) + 7 \right) = 9 \]

→ strategy: leave immediately \( l_0 \), go to \( l_3 \), and wait there 2 t.u.
Several issues on weighted timed automata

\[ g, a, C := 0 \]
Several issues on weighted timed automata

- Model-checking problems
  - reachability with an optimization criterium on the cost
  - safety with a mean-cost optimization criterium
  - model-checking WCTL, an extension of CTL with cost constraints
Several issues on weighted timed automata

- **Model-checking problems**
  - reachability with an optimization criterium on the cost
  - safety with a mean-cost optimization criterium
  - model-checking WCTL, an extension of CTL with cost constraints

- **Optimal timed games**
  - optimal reachability timed games
  - optimal mean-cost timed games
Outline

1. Introduction

2. Model-checking weighted timed automata

3. Optimal timed games

4. Conclusion
Model-checking weighted timed automata

▷ Reachability with an optimization criterium on the cost
   [Behrmann, Brinksma, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager – HSCC’01, TACAS’01, CAV’01]
   [Alur, La Torre, Pappas – HSCC’01]
   [Bouyer, Brihaye, Bruyère, Raskin – Subm. 2006]

▷ Safety with a mean-cost optimization criterium
   [Bouyer, Brinksma, Larsen – HSCC’04]

▷ Model-checking WCTL, an extension of CTL with cost constraints
   \[ A \sqcup \text{(problem } \Rightarrow A \sqcup \text{G}_{\leq 5} \text{repair) } \]
   [Brihaye, Bruyère, Raskin – FORMATS+FTRTFT’04]
   [Bouyer, Brihaye, Markey – IPL’06]
   [Bouyer, Laroussinie, Larsen, Markey, Rasmussen – 2006]
The classical region abstraction

[Diagram showing the classical region abstraction with arrows indicating transitions and labels for 'reset to 0' and 'time elapsing']
The corner-point abstraction

Idea: reduction to the discrete case

- region abstraction: not sufficient
The corner-point abstraction

**Idea:** reduction to the discrete case

- **region abstraction:** not sufficient
- **corner-point abstraction/weighted discrete graph** $\mathcal{A}_{cp}$:
The corner-point abstraction

**Idea:** reduction to the discrete case

- **region abstraction:** not sufficient
- **corner-point abstraction/weighted discrete graph** $\mathcal{A}_{cp}$:

  ![Diagram](image)

  cost rate: 3 p.u.

reset to 0
The corner-point abstraction

**Idea:** reduction to the discrete case

- **region abstraction:** not sufficient
- **corner-point abstraction/weighted discrete graph** $\mathcal{A}_{cp}$:

  ![Diagram of the corner-point abstraction with weighted discrete graph.](image)

- time elapsing
- reset to 0

- cost rate: 3 p.u.
- discrete cost: 7
The corner-point abstraction

**Idea:** reduction to the discrete case

- **region abstraction:** not sufficient
- **corner-point abstraction/weighted discrete graph** $A_{cp}$:

![Diagram of the corner-point abstraction](attachment:image.png)

- time elapsing
- reset to 0

**This abstraction is correct!**

- for computing optimal paths
- for computing optimal stationary behaviours
Optimal reachability

→ optimal reachability along a given path can be viewed as a linear programming problem

**Lemma**

Let $Z$ be a bounded zone and $f$ be a function

$$f : (t_1, \ldots, t_n) \mapsto \sum_{i=1}^{n} c_i t_i + c$$

well-defined on $\overline{Z}$. Then $\inf_Z f$ is obtained on the border of $\overline{Z}$ with integer coordinates.

Then, abstract paths in $A_{cp}$ can be approximated by real path “$\varepsilon$-close” to the abstract path
Infinite stationary behaviours: An example

A production system:

Single machine $M(D, G, P, g, p)$
Infinite stationary behaviours: An example

**A production system:**

![Diagram of a production system with states High and Low, transitions, and conditions on variables x, C, and R.]

**Question:** How to minimize

\[
\lim_{n \to +\infty} \frac{\text{accumulated cost}(n)}{\text{accumulated reward}(n)}
\]

Single machine \( M(D, G, P, g, p) \)

Operator \( O(S) \)
Infinite stationary behaviours: An example

Two machines $M_1(D = 3, P = 3, G = 4, p = 5, g = 3)$, $M_2(D = 6, P = 3, G = 2, p = 5, g = 2)$ and an Operator $O(4)$. 
Infinite stationary behaviours: An example

Two machines $M_1(D = 3, P = 3, G = 4, p = 5, g = 3)$, $M_2(D = 6, P = 3, G = 2, p = 5, g = 2)$ and an Operator $O(4)$.

$$(((H, H), x_1 = x_2 = z = 0)$$
Infinite stationary behaviours: An example

Two machines $M_1(D = 3, P = 3, G = 4, p = 5, g = 3)$,
$M_2(D = 6, P = 3, G = 2, p = 5, g = 2)$ and an Operator $O(4)$.

$18,18$ 
$((H, H), x_1 = x_2 = z = 0)$
$((L, H), x_1 = x_2 = z = 3)$
Infinite stationary behaviours: An example

Two machines $M_1(D = 3, P = 3, G = 4, p = 5, g = 3)$, $M_2(D = 6, P = 3, G = 2, p = 5, g = 2)$ and an Operator $O(4)$. 

((H, H), $x_1 = x_2 = z = 0$) 

18, 18 

((L, H), $x_1 = x_2 = z = 3$) 

8, 5 

((L, H), $x_1 = x_2 = z = 4$)
Infinite stationary behaviours: An example

Two machines $M_1(D = 3, P = 3, G = 4, p = 5, g = 3)$, $M_2(D = 6, P = 3, G = 2, p = 5, g = 2)$ and an Operator $O(4)$. 

$((H, H), x_1 = x_2 = z = 0) \xrightarrow{18,18} ((L, H), x_1 = x_2 = z = 3) \xrightarrow{8,5} ((L, H), x_1 = x_2 = z = 4) \xrightarrow{} ((H, H), x_1 = z = 0, x_2 = 4)$
Infinite stationary behaviours: An example

Two machines \( M_1(D = 3, P = 3, G = 4, p = 5, g = 3) \),
\( M_2(D = 6, P = 3, G = 2, p = 5, g = 2) \) and an Operator \( O(4) \).
Infinite stationary behaviours: An example

Two machines $M_1(D = 3, P = 3, G = 4, p = 5, g = 3)$, $M_2(D = 6, P = 3, G = 2, p = 5, g = 2)$ and an Operator $O(4)$.

\[
\text{limit } \frac{\text{cost}}{\text{reward}} = \frac{96}{66} \approx 1.455
\]
Infinite stationary behaviours: An example

Two machines $M_1(D = 3, P = 3, G = 4, p = 5, g = 3)$, $M_2(D = 6, P = 3, G = 2, p = 5, g = 2)$ and an Operator $O(4)$.

(a) Schedule with mean-cost 1,455

(b) Schedule with mean-cost 1,478
From timed to discrete behaviours (1)

- **Finite behaviours**: based on the following property

**Lemma**

Let $Z$ be a bounded zone and $f$ be a function

$$f : (t_1, ..., t_n) \mapsto \frac{\sum_{i=1}^{n} c_i t_i + c}{\sum_{i=1}^{n} r_i t_i + r}$$

well-defined on $\overline{Z}$. Then $\text{inf}_Z f$ is obtained on the border of $\overline{Z}$ with integer coordinates.
From timed to discrete behaviours (1)

- **Finite behaviours:** based on the following property

<table>
<thead>
<tr>
<th>Lemma</th>
</tr>
</thead>
<tbody>
<tr>
<td>Let $Z$ be a bounded zone and $f$ be a function $f : (t_1, ..., t_n) \mapsto \frac{\sum_{i=1}^{n} c_i t_i + c}{\sum_{i=1}^{n} r_i t_i + r}$ well-defined on $\overline{Z}$. Then $\text{inf}_Z f$ is obtained on the border of $\overline{Z}$ with integer coordinates.</td>
</tr>
</tbody>
</table>

→ for any finite path $\pi$ in $\mathcal{A}$, there exists a path $\Pi$ in $\mathcal{A}_{cp}$ such that $\text{mean-cost}(\Pi) \leq \text{mean-cost}(\pi)$

[\Pi$ is a “corner-point projection” of $\pi$]
From timed to discrete behaviours (1)

- **Finite behaviours:** based on the following property

```
Lemma
Let $Z$ be a bounded zone and $f$ be a function

$$f : (t_1, ..., t_n) \mapsto \frac{\sum_{i=1}^{n} c_i t_i + c}{\sum_{i=1}^{n} r_i t_i + r}$$

well-defined on $\overline{Z}$. Then $\inf_{Z} f$ is obtained on the border of $\overline{Z}$ with integer coordinates.
```

→ for any finite path $\pi$ in $A$, there exists a path $\Pi$ in $A_{cp}$ such that

$$\text{mean-cost}(\Pi) \leq \text{mean-cost}(\pi)$$

[$\Pi$ is a “corner-point projection” of $\pi$]

⚠️ optimal finite behaviours are not prefix-closed
From timed to discrete behaviours (2)

- **Infinite behaviours:** decompose each sufficiently long projection into cycles
From timed to discrete behaviours (2)

- **Infinite behaviours:** decompose each sufficiently long projection into cycles

The linear part will be negligible!
From timed to discrete behaviours (2)

- **Infinite behaviours:** decompose each sufficiently long projection into cycles

The linear part will be negligible!

→ the optimal cycle of $A_{cp}$ is better than any infinite path of $A$
From discrete to timed behaviours

Approximation of abstract paths:

For any path $\Pi$ of $A_{cp}$,
From discrete to timed behaviours

Approximation of abstract paths:

For any path $\Pi$ of $A_{cp}$, for any $\varepsilon > 0$,
From discrete to timed behaviours

Approximation of abstract paths:

For any path $\Pi$ of $A_{cp}$, for any $\varepsilon > 0$, there exists a path $\pi_\varepsilon$ of $A$ s.t.

$$||\Pi - \pi_\varepsilon||_\infty < \varepsilon$$
From discrete to timed behaviours

Approximation of abstract paths:

For any path Π of $A_{cp}$, for any $\varepsilon > 0$, there exists a path $\pi_\varepsilon$ of $A$ s.t.

$$||\Pi - \pi_\varepsilon||_\infty < \varepsilon$$

→ This is sufficient under the positive strongly diverging reward.
From discrete to timed behaviours

Approximation of abstract paths:

For any path $\Pi$ of $A_{cp}$, for any $\varepsilon > 0$, there exists a path $\pi_\varepsilon$ of $A$ s.t.

$$\|\Pi - \pi_\varepsilon\|_\infty < \varepsilon$$

→ This is sufficient under the positive strongly diverging reward.

For every $\eta > 0$, there exists $\varepsilon > 0$ s.t.

$$\|\Pi - \pi_\varepsilon\|_\infty < \varepsilon \Rightarrow |\text{mean-cost}(\Pi) - \text{mean-cost}(\pi_\varepsilon)| < \eta$$
Approximation of abstract paths

Diameter of a valuation:

\[ \delta(v) \]

\[ \text{"} v(z) \text{"} \quad \text{int} \quad \text{"} v(x) \text{"} \]
Approximation of abstract paths

Diameter of a valuation:

\[ \delta(v) \]

“\(v(z)\)” \(\text{int}\) “\(v(x)\)”

Computing successors:

\[ \delta(v) \]

Time elapsing

\[ \delta(v') = \delta(v) \]
Approximation of abstract paths

Diameter of a valuation:

\[ \delta(v) \]

"v(z)" \hspace{1cm} int \hspace{1cm} "v(x)"

Computing successors:

\[ \delta(v) \]

Time elapsing

\[ \delta(v') = \delta(v) \]

\[ \delta(v') = \delta(v) \]
Hypothesis: strongly non-Zeno reward

\[ y > 0, \ y := 0 \]
\[ x = 1, \ x := 0 \]

\[ \pi_{d,n}: \text{path s.t. the first transition is taken at date } d \text{ and the loop is taken } n \text{ times.} \]
Hypothesis: strongly non-Zeno reward

\[ \pi_{d,n}: \text{path s.t. the first transition is taken at date } d \text{ and the loop is taken } n \text{ times.} \]

\[ \text{reward}(\pi_{d,n}) = 2 + d \cdot n \quad \text{and} \quad \text{cost}(\pi_{d,n}) = 3 + 11d \cdot n \]
Hypothesis: strongly non-Zeno reward

\[ y > 0, \ y := 0 \quad \text{3/2} \]

\[ x = 1, \ x := 0 \quad 0/0 \]

\[ y = 1, \ y := 0 \quad 0/0 \]

\[ x = 1, \ x := 0 \quad 0/0 \]

\[ \pi_{d,n}: \text{path s.t. the first transition is taken at date } d \text{ and the loop is taken } n \text{ times.} \]

\[ \text{reward}(\pi_{d,n}) = 2 + d.n \quad \text{and} \quad \text{cost}(\pi_{d,n}) = 3 + 11d.n \]

For any real infinite path \( \pi_{d} \), \( \text{mean-cost}(\pi_{d}) = 11 \) but \( \text{mean-cost}(\pi_{0}) = \frac{3}{2} \).
Hypothesis: strongly non-Zeno reward

\[ \pi_{d,n}: \text{path s.t. the first transition is taken at date } d \text{ and the loop is taken } n \text{ times.} \]

\[ \text{reward}(\pi_{d,n}) = 2 + d \cdot n \quad \text{and} \quad \text{cost}(\pi_{d,n}) = 3 + 11d \cdot n \]

For any real infinite path \( \pi_d \), mean-cost(\( \pi_d \)) = 11 but mean-cost(\( \pi_0 \)) = \( \frac{3}{2} \).

\[ \rightarrow \text{this automaton is not strongly reward diverging} \]
Model-checking WCTL

\[ A \mathcal{G} ( \text{problem} \Rightarrow A \mathcal{G}_{\leq 5} \text{repair} ) \]

- With more than five clocks, model-checking WCTL is undecidable
  [Brihaye, Bruyère, Raskin – FORMATS+FTRTFT’04]
- With more than three clocks, model-checking WCTL is undecidable
  [Bouyer, Brihaye, Markey – IPL’06]

⇒ Short explanation at the end of the talk

- With one clock, model-checking WCTL is decidable
  [Bouyer, Laroussinie, Larsen, Markey, Rasmussen – 2006]
Outline

1. Introduction

2. Model-checking weighted timed automata

3. Optimal timed games

4. Conclusion
Decidability of timed games

<table>
<thead>
<tr>
<th>Theorem</th>
<th>[Henzinger, Kopke 1999]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Safety and reachability control in timed automata are decidable and EXPTIME-complete.</td>
<td></td>
</tr>
</tbody>
</table>

(the attractor is computable...)

→ classical regions are sufficient for solving such problems
An example

c: controllable action
u: uncontrollable action

\[
\begin{align*}
\ell_0 & \quad x \leq 2; \ c; \ y \coloneqq 0 \\
\ell_1 & \quad y = 0 \\
\ell_2 & \quad x \geq 2; \ c; \ \text{cost} = 1 \\
\ell_3 & \quad x \geq 2; \ c; \ \text{cost} = 7 \\
\text{cost}(\ell_0) & = 5 \\
\text{cost}(\ell_1) & = \text{cost}(\ell_3) = 1 \\
\end{align*}
\]
An example

c: controllable action
u: uncontrollable action

\[ \ell_0 \xrightarrow{x \leq 2; c; y := 0} \ell_1 \]
\[ \text{cost}(\ell_0) = 5 \]
\[ y = 0 \]

\[ \ell_1 \xrightarrow{u} \ell_2 \]
\[ \text{cost}(\ell_2) = 10 \]
\[ x \geq 2; c; \text{cost} = 1 \]

\[ \ell_1 \xrightarrow{u} \ell_3 \]
\[ \text{cost}(\ell_3) = 1 \]
\[ x \geq 2; c; \text{cost} = 7 \]

\[ \ell_3 \rightarrow \text{goal} \]

Question: what is the optimal cost we can ensure in state \( \ell_0 \)?
An example

c: controllable action
u: uncontrollable action

\[ \ell_0 \xrightarrow{x \leq 2; \, c; \, y := 0} \ell_1 \]
\[ \text{cost}(\ell_0) = 5 \quad y = 0 \]

\[ \ell_1 \xrightarrow{u} \ell_2 \]
\[ \text{cost}(\ell_2) = 10 \]
\[ x \geq 2; \, c; \, \text{cost} = 1 \]

\[ \ell_1 \xrightarrow{u} \ell_3 \]
\[ \text{cost}(\ell_3) = 1 \]
\[ x \geq 2; \, c; \, \text{cost} = 7 \]

\[ \ell_3 \rightarrow \text{goal} \]

**Question:** what is the optimal cost we can ensure in state \( \ell_0 \)?

\[ 5t + 10(2 - t) + 1 \]
An example

\(c\): controllable action  
\(u\): uncontrollable action

\[
\begin{align*}
\ell_0 & \xrightarrow{x \leq 2; c; y := 0} \ell_1 \\
\text{cost}(\ell_0) &= 5 & y = 0 && \ell_1 \xrightarrow{u} \ell_2 \\
\text{cost}(\ell_1) &= & \ell_2 & \xrightarrow{x \geq 2; c; \text{cost} = 1} \ell_3 \\
\text{cost}(\ell_2) &= 10 & \ell_3 & \xrightarrow{x \geq 2; c; \text{cost} = 7} \ell_0 \\
\text{cost}(\ell_3) &= 1 \\
\end{align*}
\]

**Question:** what is the optimal cost we can ensure in state \(\ell_0\)?

\[
5t + 10(2 - t) + 1 , \ 5t + (2 - t) + 7
\]
**An example**

- **c**: controllable action
- **u**: uncontrollable action

The diagram represents a timed automaton with transitions labeled with conditions and costs.

- **ℓ₀**: Initial state with cost $\text{cost}(ℓ₀) = 5$ and transition to **ℓ₁** with condition $x \leq 2; c; y := 0$.
- **ℓ₁**: Transition to **ℓ₂** with condition $y = 0$.
- **ℓ₂**: Transition to **ℓ₃** with condition $x \geq 2; c; \text{cost} = 1$.
- **ℓ₃**: Transition to a goal state with condition $x \geq 2; c; \text{cost} = 7$.

**cost(ℓ₂) = 10**

**Question**: What is the optimal cost we can ensure in state ℓ₀?

$$\max (5t + 10(2 - t) + 1, 5t + (2 - t) + 7)$$
An example

$c$: controllable action
$u$: uncontrollable action

cost($\ell_0$) = 5

$x \leq 2; c; y := 0$

cost($\ell_1$) = cost($\ell_2$) = 10

$y = 0$

$u$

$x \geq 2; c; cost = 1$

$\ell_3$

cost($\ell_3$) = 1

$u$

$x \geq 2; c; cost = 7$

Question: what is the optimal cost we can ensure in state $\ell_0$?

$$\inf_{0 \leq t \leq 2} \max \left( 5t + 10(2 - t) + 1 , 5t + (2 - t) + 7 \right) = 14 + \frac{1}{3}$$
An example

c: controllable action
u: uncontrollable action

\[
\ell_0 \quad x \leq 2; \ c; \ y := 0
\]

\[
\ell_0 \quad \text{cost}(\ell_0) = 5
\]

\[
\ell_1 \quad y = 0
\]

\[
\ell_1 \quad \text{cost}(\ell_1) = 10
\]

\[
\ell_2 \quad x \geq 2; \ c; \ \text{cost} = 1
\]

\[
\ell_3 \quad x \geq 2; \ c; \ \text{cost} = 7
\]

\[
\ell_3 \quad \text{cost}(\ell_3) = 1
\]

Question: what is the optimal cost we can ensure in state \( \ell_0 \)?

\[
\inf_{0 \leq t \leq 2} \max \left( 5t + 10(2 - t) + 1, 5t + (2 - t) + 7 \right) = 14 + \frac{1}{3}
\]

\( \Rightarrow \) strategy: wait in \( \ell_0 \), and when \( t = \frac{4}{3} \), go to \( \ell_1 \)
An example

c: controllable action
u: uncontrollable action

**Question:** what is the optimal cost we can ensure in state $\ell_0$?

$$\inf_{0 \leq t \leq 2} \max (5t + 10(2-t) + 1, 5t + (2-t) + 7) = 14 + \frac{1}{3}$$

→ strategy: wait in $\ell_0$, and when $t = \frac{4}{3}$, go to $\ell_1$

► How to automatically compute such optimal costs?
An example

c: controllable action
u: uncontrollable action

\[ \ell_0 \xrightarrow{x \leq 2; \ c} \ell_1 \xrightarrow{y = 0} \ell_2 \]

\[ \text{cost}(\ell_0) = 5 \]

\[ \ell_2 \quad x \geq 2; \ c; \ \text{cost} = 1 \]

\[ \ell_3 \quad x \geq 2; \ c; \ \text{cost} = 7 \]

\[ \text{cost}(\ell_3) = 1 \]

\[ \ell_1 \xrightarrow{u} \ell_2 \]

\[ \ell_1 \xrightarrow{u} \ell_3 \]

Question: what is the optimal cost we can ensure in state \( \ell_0 \)?

\[ \inf_{0 \leq t \leq 2} \max \left( 5t + 10(2 - t) + 1, \ 5t + (2 - t) + 7 \right) = 14 + \frac{1}{3} \]

\[ \Rightarrow \text{strategy: wait in } \ell_0, \text{ and when } t = \frac{4}{3}, \text{ go to } \ell_1 \]

\[ \Rightarrow \text{ How to automatically compute such optimal costs?} \]

\[ \Rightarrow \text{ How to synthesize optimal strategies (if one exists)?} \]
A hot topic!

- [Asarin, Maler – HSCC’99]:
  - optimal time is computable in timed games
A hot topic!

- [Asarin, Maler – HSCC’99]:
  - optimal time is computable in timed games
- [La Torre, Mukhopadhyay, Murano – TCS@02]:
  - case of acyclic games
A hot topic!

- [Asarin, Maler – HSCC’99]:
  - optimal time is computable in timed games
- [La Torre, Mukhopadhyay, Murano – TCS@02]:
  - case of acyclic games
- [Alur, Bernadsky, Madhusudan – ICALP’04]:
  - complexity of $k$-step games
  - under a strongly non-Zeno assumption, optimal cost is computable
A hot topic!

- [Asarin, Maler – HSCC’99]:
  - optimal time is computable in timed games

- [La Torre, Mukhopadhyay, Murano – TCS@02]:
  - case of acyclic games

- [Alur, Bernadsky, Madhusudan – ICALP’04]:
  - complexity of $k$-step games
  - under a strongly non-Zeno assumption, optimal cost is computable

- [Bouyer, Cassez, Fleury, Larsen – FSTTCS’04]:
  - structural properties of strategies (e.g. memory)
  - under a strongly non-Zeno assumption, optimal cost is computable
A hot topic!

- [Asarin, Maler – HSCC’99]:
  - optimal time is computable in timed games
- [La Torre, Mukhopadhyay, Murano – TCS@02]:
  - case of acyclic games
- [Alur, Bernadsky, Madhusudan – ICALP’04]:
  - complexity of $k$-step games
  - under a strongly non-Zeno assumption, optimal cost is computable
- [Bouyer, Cassez, Fleury, Larsen – FSTTCS’04]:
  - structural properties of strategies (e.g. memory)
  - under a strongly non-Zeno assumption, optimal cost is computable
- [Brihaye, Bruyère, Raskin – FORMATS’05]:
  - with five clocks, optimal cost is not computable!
  - with one clock and one stopwatch cost, optimal cost is computable
A hot topic!

- [Asarin, Maler – HSCC’99]:
  - optimal time is computable in timed games
- [La Torre, Mukhopadhyay, Murano – TCS@02]:
  - case of acyclic games
- [Alur, Bernadsky, Madhusudan – ICALP’04]:
  - complexity of $k$-step games
  - under a strongly non-Zeno assumption, optimal cost is computable
- [Bouyer, Cassez, Fleury, Larsen – FSTTCS’04]:
  - structural properties of strategies (e.g. memory)
  - under a strongly non-Zeno assumption, optimal cost is computable
- [Brihaye, Bruyère, Raskin – FORMATS’05]:
  - with five clocks, optimal cost is not computable!
  - with one clock and one stopwatch cost, optimal cost is computable
- [Bouyer, Brihaye, Markey – IPL’06]:
  - with three clocks, optimal cost is not computable
A hot topic!

- [Asarin, Maler – HSCC’99]:
  - optimal time is computable in timed games
- [La Torre, Mukhopadhyay, Murano – TCS@02]:
  - case of acyclic games
- [Alur, Bernadsky, Madhusudan – ICALP’04]:
  - complexity of k-step games
  - under a strongly non-Zeno assumption, optimal cost is computable
- [Bouyer, Cassez, Fleury, Larsen – FSTTCS’04]:
  - structural properties of strategies (e.g. memory)
  - under a strongly non-Zeno assumption, optimal cost is computable
- [Brihaye, Bruyère, Raskin – FORMATS’05]:
  - with five clocks, optimal cost is not computable!
  - with one clock and one stopwatch cost, optimal cost is computable
- [Bouyer, Brihaye, Markey – IPL’06]:
  - with three clocks, optimal cost is not computable
- [Bouyer, Larsen, Markey, Rasmussen – Subm.’06]:
  - with one clock, optimal cost is computable
A hot topic!

- [Asarin, Maler – HSCC’99]:
  - optimal time is computable in timed games
- [La Torre, Mukhopadhyay, Murano – TCS@02]:
  - case of acyclic games
- [Alur, Bernadsky, Madhusudan – ICALP’04]:
  - complexity of $k$-step games
  - under a strongly non-Zeno assumption, optimal cost is computable
- [Bouyer, Cassez, Fleury, Larsen – FSTTCS’04]:
  - structural properties of strategies (e.g. memory)
  - under a strongly non-Zeno assumption, optimal cost is computable
- [Brihaye, Bruyère, Raskin – FORMATS’05]:
  - with five clocks, optimal cost is not computable!
  - with one clock and one stopwatch cost, optimal cost is computable
- [Bouyer, Brihaye, Markey – IPL’06]:
  - with three clocks, optimal cost is not computable
- [Bouyer, Larsen, Markey, Rasmussen – Subm.’06]:
  - with one clock, optimal cost is computable

See Kim’s talk
Do optimal strategies always exist?

\[
\begin{align*}
\ell_0 & \quad \text{cost } 1 \quad x < 1; \ c \\
\ell_1 & \quad \text{cost } 2 \quad x = 1; \ c \\
W &
\end{align*}
\]

\[
\begin{align*}
f(\ell_0, x < 1) &= \lambda \\
f(\ell_1, x < 1) &= \lambda \\
f(\ell_1, x = 1) &= c
\end{align*}
\]
Do optimal strategies always exist?

\[
\begin{align*}
\ell_0 & \quad \text{cost} = 1 \\
\ell_1 & \quad \text{cost} = 2 \\
W & \\
\end{align*}
\]

\[
\begin{align*}
x < 1 & \quad x < 1; c \\
x < 1 & \quad x = 1; c \\
\end{align*}
\]

\[
\begin{align*}
f(\ell_0, x < 1) &= \lambda \\
f(\ell_0, x < 1) &= \lambda \\
f(\ell_1, x < 1) &= \lambda \\
f(\ell_1, x = 1) &= c \\
f(\ell_1, x = 1) &= c \\
\end{align*}
\]

\[
\begin{align*}
f_\varepsilon(\ell_0, x < 1 - \varepsilon) &= \lambda \\
f_\varepsilon(\ell_0, 1 - \varepsilon \leq x < 1) &= c \\
f_\varepsilon(\ell_1, x < 1) &= \lambda \\
f_\varepsilon(\ell_1, x = 1) &= c \\
\end{align*}
\]
Do optimal strategies always exist?

\[
\begin{align*}
&\text{cost} = 1 & &\text{cost} = 2 \\
&x < 1; \ c & &x = 1; \ c \\
\end{align*}
\]

\[
\begin{align*}
&f(\ell_0, x < 1) = \lambda \\
&f(\ell_1, x < 1) = \lambda \\
&f(\ell_1, x = 1) = c \\
&f(\ell_0, x < 1 - \varepsilon) = \lambda \\
&f(\ell_0, 1 - \varepsilon \leq x < 1) = c \\
&f(\ell_1, x < 1) = \lambda \\
&f(\ell_1, x = 1) = c
\end{align*}
\]

→ no optimal strategy exists, but rather a family \((f_\varepsilon)_{\varepsilon > 0}\) of \(\varepsilon\)-approximating strategies \((\text{cost}(f_\varepsilon) = 1 + \varepsilon)\)
An encoding (1)

Idea: transform the cost into a decreasing linear hybrid variable

\[ G \]

\[ G' \]

\[ \ell_0 \quad g, \ a, \ Y := 0 \quad \ell_1 \quad \text{cost} = 1 \]

\[ \ell_0' \quad g, \ a, \ Y := 0 \quad \ell_1' \quad \text{cost} := \text{cost} - 1 \]

Winning: \( W \)

Winning: \( W \land \text{cost} \geq 0 \)
An encoding (1)

Idea: transform the cost into a decreasing linear hybrid variable

\[ G \quad g, \ a, \ Y := 0 \quad \text{cost} = 1 \quad \Rightarrow \quad G' \quad g, \ a, \ Y := 0 \quad \text{cost} := \text{cost} - 1 \]

Winning: \( W \)

Winning: \( W \land \text{cost} \geq 0 \)

Theorem

For priced timed games (under some hypotheses),

\[ \exists f \text{ winning strategy in } G \quad \text{s.t. } \text{cost}(f, (\ell, v)) \leq \gamma \quad \iff \quad (\ell, v, \text{cost} = \gamma) \text{ winning in } G' \]

+ constructive proof
An encoding (2)

The set of winning states in $G'$ is upward-closed for the cost, \textit{i.e.} of the form

$$\bigcup_{i \in l} (P_i \land \text{cost} \succ_i k_i) \quad \text{(with } \succ_i \text{ either } > \text{ or } \geq\text{)}$$
An encoding (2)

The set of winning states in $G'$ is upward-closed for the cost, i.e. of the form

$$\bigcup_{i \in I} (P_i \land \text{cost} \succ_i k_i) \quad \text{(with } \succ_i \text{ either } > \text{ or } \geq)$$

Corollary

For priced timed games (under some hypotheses),

- "reachable" optimal cost, or not (cost $\geq \gamma$ or cost $> \gamma$)
- existence of an optimal strategy decidable

+ constructive proof
An encoding (2)

The set of winning states in $G'$ is upward-closed for the cost, i.e. of the form

$$
\bigcup_{i \in I} (P_i \land \text{cost } \succ_i k_i) \quad \text{(with } \succ_i \text{ either } > \text{ or } \geq)
$$

**Corollary**

For priced timed games (under some hypotheses),

- “reachable” optimal cost, or not (cost $\geq \gamma$ or cost $> \gamma$)
- existence of an optimal strategy decidable

+ constructive proof

**Nature of the strategy:**

- state-based for the hybrid game, thus **cost-dependent** for the timed game
- cost-dependence is unavoidable in general!
- cost-independent strategies for syntactical restrictions of the games
  
  $c$: large constraints,  
  $u$: strict constraints
Memoryless strategies are not powerful enough

- optimal cost: 2
- optimal strategy:
Memoryless strategies are not powerful enough

- optimal cost: 2
- optimal strategy: if \( d \) is the time before a \( u \) occurs, and \( d' \) is the time waited in \( \ell_1 \), the cost of the run is \( 2d + d' \).
Memoryless strategies are not powerful enough

**optimal cost:** 2

**optimal strategy:** if $d$ is the time before a $u$ occurs, and $d'$ is the time waited in $l_1$, the cost of the run is $2d + d'$.

$$2d + d' \leq 2$$
Memoryless strategies are not powerful enough

\[ \ell_0 \xrightarrow{x \leq 1} \ell_1 \xrightarrow{x < 1, u, x, y := 0} W \xrightarrow{y > 0, c} \]

- **optimal cost**: 2
- **optimal strategy**: if \( d \) is the time before a \( u \) occurs, and \( d' \) is the time waited in \( \ell_1 \), the cost of the run is \( 2d + d' \).

\[ 2d + d' \leq 2 \]

\((\text{accumulated cost}) + d' \leq 2\)
Hypotheses for termination

- all clocks are bounded (not restrictive)
- the cost function is *strictly non-Zeno*
  → This condition is restrictive, but is decidable
Hypotheses for termination

- all clocks are bounded (not restrictive)
- the cost function is strictly non-Zeno
  → This condition is restrictive, but is decidable
Undecidability – Shape of the reduction

Original reduction: [Brihaye, Bruyère, Raskin – FORMATS’05]
This reduction: [Bouyer, Brihaye, Markey – IPL’06]
Undecidability – Shape of the reduction

Original reduction: [Brihaye, Bruyère, Raskin – FORMATS’05]
This reduction: [Bouyer, Brihaye, Markey – IPL’06]

Simulation of a two-counter machine:

- **player 1** simulates the two-counter machine
- **player 2** checks that **player 1** does not cheat
Undecidability – Shape of the reduction

Original reduction: [Brihaye, Bruyère, Raskin – FORMATS’05]
This reduction: [Bouyer, Brihaye, Markey – IPL’06]

Simulation of a two-counter machine:
- player 1 simulates the two-counter machine
- player 2 checks that player 1 does not cheat

Encoding of the counters:
- counter $c_1$ is encoded by a clock $x_1$ s.t. $x_1 = \frac{1}{2c_1}$
- counter $c_2$ is encoded by a clock $x_2$ s.t. $x_2 = \frac{1}{3c_2}$
- $x_1$ and $x_2$ will be alternatively $x$, $y$ or $z$
Undecidability – Shape of the reduction

Original reduction: [Brihaye, Bruyère, Raskin – FORMATS’05]
This reduction: [Bouyer, Brihaye, Markey – IPL’06]

Simulation of a two-counter machine:
- player 1 simulates the two-counter machine
- player 2 checks that player 1 does not cheat

Encoding of the counters:
- counter $c_1$ is encoded by a clock $x_1$ s.t. $x_1 = \frac{1}{2^{c_1}}$
- counter $c_2$ is encoded by a clock $x_2$ s.t. $x_2 = \frac{1}{3^{c_2}}$
- $x_1$ and $x_2$ will be alternatively $x$, $y$ or $z$

The aim of player 1 is to win (reach a $W$-state) with cost $\leq 3$, 
Undecidability – Shape of the reduction

Original reduction: [Brihaye, Bruyère, Raskin – FORMATS’05]
This reduction: [Bouyer, Brihaye, Markey – IPL’06]

Simulation of a two-counter machine:
- player 1 simulates the two-counter machine
- player 2 checks that player 1 does not cheat

Encoding of the counters:
- counter $c_1$ is encoded by a clock $x_1$ s.t. $x_1 = \frac{1}{2c_1}$
- counter $c_2$ is encoded by a clock $x_2$ s.t. $x_2 = \frac{1}{3c_2}$
- $x_1$ and $x_2$ will be alternatively $x$, $y$, or $z$

The aim of player 1 is to win (reach a $W$-state) with cost $\leq 3$, and

Player 1 has a winning strategy with cost $\leq 3$
iff
the two-counter machine halts
Simulation of an incrementation

Instruction $i$: $c_1++$; goto instruction $j$

$$\begin{cases} x = \frac{1}{2}c_1 \\ y = \frac{1}{3}c_2 \end{cases}$$
Simulation of an incrementation

Instruction $i$: $c_1 + +$; goto instruction $j$

\[ y = 1, y := 0 \]

\[ \begin{align*}
 c_1 & \mapsto x \\
 c_2 & \mapsto y \\
\end{align*} \]

\[ \begin{cases}
 x = \frac{1}{2c_1} \\
 y = \frac{1}{3c_2}
\end{cases} \]

\[ \begin{align*}
 c_1 & \mapsto z \\
 c_2 & \mapsto y \\
\end{align*} \]

\[ \begin{cases}
 z = \frac{1}{2c_1 + 1} \\
 y = \frac{1}{3c_2}
\end{cases} \]
Adding $x$ or $1 - x$ to the cost variable

The cost is increased by $x_0$
Adding $x$ or $1 - x$ to the cost variable

The cost is increased by $x_0$

The cost is increased by $1 - x_0$
Adding $x$ or $1 - x$ to the cost variable

$\text{Add}^+(x, \{z\})$

$\text{Add}^-(x, \{z\})$

The cost is increased by $x_0$

The cost is increased by $1 - x_0$
Checking $y = 2x$

In $W_1$, cost = $2x_0 + (1 - y_0) + 2$.

In $W_2$, cost = $2(1 - x_0) + y_0 + 1$. 
Checking $y = 2x$

\[ \text{In } W_1, \text{ cost } = 2x_0 + (1 - y_0) + 2. \]
\[ \text{In } W_2, \text{ cost } = 2(1 - x_0) + y_0 + 1. \]

- if $y_0 < 2x_0$, player 2 chooses the first branch: in $W_1$, cost $> 3$
Checking $y = 2x$

In $W_1$, cost $= 2x_0 + (1 - y_0) + 2$.
In $W_2$, cost $= 2(1 - x_0) + y_0 + 1$.

- if $y_0 < 2x_0$, player 2 chooses the first branch: in $W_1$, cost $> 3$
- if $y_0 > 2x_0$, player 2 chooses the second branch: in $W_2$, cost $> 3$
Checking $y = 2x$

\[
\begin{align*}
&\text{In } W_1, \text{ cost } = 2x_0 + (1 - y_0) + 2. \\
&\text{In } W_2, \text{ cost } = 2(1 - x_0) + y_0 + 1.
\end{align*}
\]

- if $y_0 < 2x_0$, player 2 chooses the first branch: in $W_1$, cost $> 3$
- if $y_0 > 2x_0$, player 2 chooses the second branch: in $W_2$, cost $> 3$
- if $y_0 = 2x_0$, in $W_1$ or in $W_2$, cost $= 3$. 

Optimal timed games
How to get rid of tick clock $u$?

Test($x = 2z$, \{y\})
How to get rid of tick clock $u$?

We will ensure that:
- no cost is accumulated in $D$-states
How to get rid of tick clock $u$?

We will ensure that:

-$\triangleright$ no cost is accumulated in $D$-states
-$\triangleright$ the delay between the $A$-state and the $D$-state is 1 t.u.
How to get rid of tick clock $u$?

We will ensure that:

- no cost is accumulated in $D$-states
- the delay between the $A$-state and the $D$-state is 1 t.u.
  - the value of $x$ in $D$ is of the form $\frac{1}{2^n}$

Test($x = 2z, \{y\}$)
How to get rid of tick clock $u$?

We will ensure that:

- no cost is accumulated in $D$-states
- the delay between the $A$-state and the $D$-state is 1 t.u.
  - the value of $x$ in $D$ is of the form $\frac{1}{2^n}$
  - the value of $y$ in $D$ is of the form $\frac{1}{3^m}$

Cost = 3

Halt
How to get rid of tick clock $u$?

We will ensure that:

- no cost is accumulated in $D$-states
- the delay between the $A$-state and the $D$-state is 1 t.u.
  - the value of $x$ in $D$ is of the form $\frac{1}{2^n}$
  - the value of $y$ in $D$ is of the form $\frac{1}{3^m}$
Checking that $x$ is of the form $\frac{1}{2^n}$

Test($y = 2x, \{z\}$)
Extension to undecidability of WCTL

We build the same automaton $\mathcal{A}_M$, and prove that:

the two-counter machine $\mathcal{M}$ halts iff $\mathcal{A}_M \models \Phi$

where

$$\Phi \equiv \mathbf{E} (D \rightarrow \varphi) \mathbf{U} \leq_0 \text{Halt}$$

with $\varphi \equiv \bigwedge_{i=1,2,3} \mathbf{E} (D \mathbf{U} \leq_0 \varphi_i)$

$$\varphi_1 \equiv S \wedge \mathbf{E} \mathbf{F} \leq_1 T \wedge \mathbf{E} \mathbf{F} \geq_1 T$$ evaluated in $\text{Test}(x = 2z, \{y\})$

$$\varphi_2 \equiv P_2 \wedge \mathbf{E} ((Q_2 \rightarrow \mathbf{E} (Q_2 \mathbf{U} \varphi_1)) \mathbf{U} R_2)$$ evaluated in $\text{Power}_2(x, \{y, z\})$
Outline

1. Introduction

2. Model-checking weighted timed automata

3. Optimal timed games

4. Conclusion
Conclusion and further work

Model-checking

- “basic” properties are decidable
- efficient symbolic computations have even been proposed
  ➔ implemented in tool Uppaal Cora
- branching-time properties are undecidable
Conclusion and further work

**Model-checking**
- “basic” properties are decidable
- efficient symbolic computations have even been proposed → implemented in tool Uppaal Cora
- branching-time properties are undecidable
- what about linear-time properties?
- consider more general cost functions
Conclusion and further work

Model-checking

- “basic” properties are decidable
- efficient symbolic computations have even been proposed ➔ implemented in tool Uppaal Cora
- branching-time properties are undecidable
- what about linear-time properties?
- consider more general cost functions

Optimal timed games

- optimal cost is in general not computable in timed games
- under some assumption, it becomes computable
- complexity issues and properties of strategies have also been studied
Conclusion and further work

**Model-checking**

- “basic” properties are decidable
- efficient symbolic computations have even been proposed
  → implemented in tool Uppaal Cora
- branching-time properties are undecidable
- what about linear-time properties?
- consider more general cost functions

**Optimal timed games**

- optimal cost is in general not computable in timed games
- under some assumption, it becomes computable
- complexity issues and properties of strategies have also been studied
- investigate further mean-cost optimal timed games
- approximate optimal cost
- propose more algorithmics solutions
- o-minimal optimal timed games
- ...