Quantitative timed games

Patricia Bouyer

LSV – CNRS & ENS Cachan – France

Based on joint works with Thomas Brihaye, Véronique Bruyère,
Uli Fahrenberg, Kim G. Larsen, Nicolas Markey,
Jean-François Raskin, Jirí Srba, and Jacob Illum Rasmussen
Outline

1. Introduction

2. Modelling and optimizing resources in timed systems

3. Managing resources

4. Conclusion
A starting example
Natural questions

- Can I reach Pontivy from Oxford?
- What is the **minimal time** to reach Pontivy from Oxford?
- What is the **minimal fuel consumption** to reach Pontivy from Oxford?
- What if there is an **unexpected event**?
- Can I use my computer all the way?
A first model of the system
Can I reach Pontivy from Oxford?

This is a reachability question in a finite graph: Yes, I can!
A second model of the system
How long will that take?

It is a reachability (and optimization) question in a timed automaton: at least 350mn = 5h50mn!
An example of a timed automaton

```plaintext
An example of a timed automaton

Introduction

An example of a timed automaton

```

```

```plaintext
An example of a timed automaton

```
An example of a timed automaton

```plaintext
x := 0, x ≤ 15
y := 0, 15 ≤ x ≤ 16
delayed
y := 0

done, 22 ≤ y ≤ 25

problem

repair, x ≤ 15
y := 0

repair
2 ≤ y ∧ x ≤ 56
y := 0

repairing

failsafe
```

```
X 0
Y 0
```

```plaintext
safe
```
An example of a timed automaton

\[
\begin{array}{c}
\text{safe} \quad \xrightarrow{23} \quad \text{safe} \\
X & 0 & 23 \\
Y & 0 & 23 \\
\end{array}
\]
An example of a timed automaton

\[
\begin{align*}
\text{safe} & \xrightarrow{23} \text{safe} \quad \text{problem, } x:=0 \\
X & = 0 \quad 23 \quad 0 \\
y & = 0 \quad 23 \quad 23
\end{align*}
\]
An example of a timed automaton

\[
\begin{align*}
\text{problem, } x &= 0 \\
\text{safe} &\rightarrow \text{alarm} \\
\text{repair, } x &\leq 15 \\
\text{delayed, } y &\leq 0 \\
\text{failsafe} &\rightarrow \\
\end{align*}
\]

\[
\begin{align*}
\text{done, } 22 &\leq y \leq 25 \\
\text{repair, } x &\leq 15 \\
\text{repair, } y &\leq 0 \\
\text{repair, } 2 \leq y \land x \leq 56 \\
\end{align*}
\]
An example of a timed automaton

![Timed Automaton Diagram]

- **Safe**
  - \( x := 0 \)
  - \( y := 0 \)
  - \( 15 \leq x \leq 16 \)
  - \( 2 \leq y \land x \leq 56 \)
  - \( y := 0 \)
- **Alarm**
  - \( x := 0 \)
  - \( y := 0 \)
  - \( 15 \leq y \leq 25 \)
- **Repairing**
  - \( 2 \leq y \land x \leq 56 \)
  - \( y := 0 \)
- **Failsafe**
  - \( y := 0 \)

Transition Table:

<table>
<thead>
<tr>
<th>State</th>
<th>Action</th>
<th>Next State</th>
</tr>
</thead>
<tbody>
<tr>
<td>Safe</td>
<td>23</td>
<td>Safe</td>
</tr>
<tr>
<td></td>
<td>Problem</td>
<td>Alarm</td>
</tr>
<tr>
<td>Alarm</td>
<td>15.6</td>
<td>Alarm</td>
</tr>
<tr>
<td></td>
<td>Delayed</td>
<td>Failsafe</td>
</tr>
</tbody>
</table>

- **Safe** \( x := 0 \), \( y := 0 \)
- **Failsafe** \( y := 0 \)
- **Alarm** \( x := 0 \), \( y := 0 \)
- **Repairing** \( x := 0 \), \( y := 0 \)

\( x, y \) in states:

- Safe: \( x = 0 \), \( y = 0 \)
- Alarm: \( x = 0 \), \( y = 0 \)
- Repairing: \( x = 0 \), \( y = 0 \)
- Failsafe: \( y = 0 \)

**Formalizing the Problem**

- \( x := 0 \)
- \( y := 0 \)
- \( 15 \leq x \leq 16 \)
- \( 2 \leq y \land x \leq 56 \)
- \( y := 0 \)

**Example Transition**

- \( x := 0 \)
- \( y := 0 \)
- \( 15 \leq x \leq 16 \)
- \( 2 \leq y \land x \leq 56 \)
- \( y := 0 \)
An example of a timed automaton

---

**Introduction**

An example of a timed automaton

---

**Diagram**

---

**Table**

<table>
<thead>
<tr>
<th>State</th>
<th>Transition</th>
<th>Next State</th>
<th>Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>safe</td>
<td>23</td>
<td>safe</td>
<td>x:=0</td>
</tr>
<tr>
<td>problem</td>
<td></td>
<td>alarm</td>
<td>y:=0</td>
</tr>
<tr>
<td>alarm</td>
<td>15.6</td>
<td>alarm</td>
<td>y:=0</td>
</tr>
<tr>
<td>repaired</td>
<td></td>
<td>repaired</td>
<td>y:=0</td>
</tr>
<tr>
<td>done</td>
<td></td>
<td>done</td>
<td>y:=0</td>
</tr>
<tr>
<td>failsafe</td>
<td>2.3</td>
<td>failsafe</td>
<td>y:=0</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>failsafe</td>
<td>y:=0</td>
</tr>
</tbody>
</table>

---

**Equations**

\[ x(0) = 0 \]
\[ y(0) = 0 \]

---

**Legend**

- **safe**
- **problem**
- **alarm**
- **repaired**
- **done**
- **failsafe**
An example of a timed automaton

```
<table>
<thead>
<tr>
<th></th>
<th>23</th>
<th></th>
<th>problem</th>
<th>15.6</th>
<th></th>
<th>delayed</th>
<th>failsafe</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>0</td>
<td>23</td>
<td>0</td>
<td>15.6</td>
<td></td>
<td></td>
<td>15.6</td>
</tr>
<tr>
<td>y</td>
<td>0</td>
<td>23</td>
<td>23</td>
<td>38.6</td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>...</td>
<td>15.6</td>
<td></td>
<td>17.9</td>
<td>17.9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>0</td>
<td>2.3</td>
<td>17.9</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
An example of a timed automaton
An example of a timed automaton
Timed automata

**Theorem [AD90,CY92]**

The (time-optimal) reachability problem is decidable (and PSPACE-complete) for timed automata.

---

[AD90] Alur, Dill. Automata for modeling real-time systems (ICALP’90).

Timed automata

Theorem [AD90,CY92]

The (time-optimal) reachability problem is decidable (and PSPACE-complete) for timed automata.

[AD90] Alur, Dill. Automata for modeling real-time systems (ICALP’90).
The region abstraction
The region abstraction

- “compatibility” between regions and constraints
The region abstraction

- “compatibility” between regions and constraints
- “compatibility” between regions and time elapsing
The region abstraction

- “compatibility” between regions and constraints
- “compatibility” between regions and time elapsing
The region abstraction

- “compatibility” between regions and constraints
- “compatibility” between regions and time elapsing

\[ \leadsto \] an equivalence of finite index

\[ \sim \] a time-abstract bisimulation
The region abstraction

---

\[\text{time elapsing}\]

\[\text{reset to 0}\]
Outline

1. Introduction

2. Modelling and optimizing resources in timed systems

3. Managing resources

4. Conclusion
Modelling and optimizing resources in timed systems

- System resources might be relevant and even crucial information
Modelling resources in timed systems

- System resources might be relevant and even crucial information
  - energy consumption,
  - memory usage,
  - price to pay,
  - bandwidth,
  - ...

Modelling and optimizing resources in timed systems
Modelling resources in timed systems

- System resources might be relevant and even crucial information
  - energy consumption,
  - memory usage,
  - price to pay,
  - bandwidth,
  - ...

  \( \Rightarrow \) timed automata are not powerful enough!
Modelling resources in timed systems

- System resources might be relevant and even crucial information
  - energy consumption,
  - memory usage,
  - price to pay,
  - bandwidth,
  - ...

  $\Rightarrow$ timed automata are not powerful enough!

- A possible solution: use hybrid automata
Modelling resources in timed systems

- System resources might be relevant and even crucial information
  - energy consumption,
  - memory usage,
  - price to pay,
  - bandwidth,
  - ... 

\[ \bar{T} = -0.5T \quad (T \geq 18) \]
\[ \bar{T} = 2.25 - 0.5T \quad (T \leq 22) \]

\[ T \leq 19 \]
\[ T \geq 21 \]

\[ \Rightarrow \] timed automata are not powerful enough!

- A possible solution: use hybrid automata

The thermostat example

- Weighted/priced timed automata

Theorem [HKPV95]
The reachability problem is undecidable in hybrid automata.
Modelling resources in timed systems

- System resources might be relevant and even crucial information
  - energy consumption,
  - memory usage,
  - price to pay,
  - bandwidth,
  - ...

  \(\leadsto\) timed automata are not powerful enough!

- A possible solution: use hybrid automata

The thermostat example

\[
\begin{align*}
\text{Off:} & \quad \dot{T} = -0.5T \\
& \quad (T \geq 18) \\
\text{On:} & \quad \dot{T} = 2.25 - 0.5T \\
& \quad (T \leq 22)
\end{align*}
\]
Modelling resources in timed systems

- System resources might be relevant and even crucial information:
  - energy consumption,
  - memory usage,
  - price to pay,
  - bandwidth,
  - ...

  \(\Rightarrow\) timed automata are not powerful enough!

- A possible solution: use hybrid automata

**Theorem [HKPV95]**

The reachability problem is **undecidable** in hybrid automata.

[HKPV95] Henzinger, Kopke, Puri, Varaiya. What’s decidable wbout hybrid automata? *(SToC’95).*
Modelling resources in timed systems

- System resources might be relevant and even crucial information
  - energy consumption,
  - memory usage,
  - price to pay,
  - bandwidth,
  - ...

  \[ \rightarrow \] timed automata are not powerful enough!

- A possible solution: use hybrid automata

**Theorem [HKPV95]**

The reachability problem is **undecidable** in hybrid automata.

- An alternative: weighted/priced timed automata [ALP01,BFH+01]

  \[ \rightarrow \] hybrid variables do not constrain the system


A third model of the system
How much fuel will I use?

It is a quantitative (optimization) problem in a priced/weighted timed automaton: at least 68 anti-planet units!
Weighted/priced timed automata [ALP01,BFH+01]

\[
\begin{align*}
\ell_0 & \xrightarrow{x \leq 2,c,y := 0} \ell_1 \\
\ell_1 & \xrightarrow{u} \ell_2 \xrightarrow{x = 2,c} \ell_3 \xrightarrow{+1} \text{smiley} \\
\ell_2 & \xrightarrow{u} \ell_1 \\
\ell_3 & \xrightarrow{x = 2,c} \ell_2
\end{align*}
\]

\[\ell_0 + 5, \ell_1, \ell_2, \ell_3, +10, +1\]

Weighted/priced timed automata \[\text{[ALP01,BFH+01]}\]

\[
\ell_0 \xrightarrow{1.3} \ell_0 \xrightarrow{c} \ell_1 \xrightarrow{u} \ell_3 \xrightarrow{0.7} \ell_3 \xrightarrow{c} \text{smiley}
\]

\[
x \quad 0 \quad 1.3 \quad 1.3 \quad 1.3 \quad 2
\]

\[
y \quad 0 \quad 1.3 \quad 0 \quad 0 \quad 0.7
\]

Weighted/priced timed automata [ALP01,BFH+01]

\[ x \leq 2, c, y := 0 \]

\[
\begin{align*}
\ell_0 & \xrightarrow{+5} \ell_0 \\
\ell_1 & \xrightarrow{u} \ell_1 \\
\ell_2 & \xrightarrow{x = 2, c} \ell_2 \\
\ell_3 & \xrightarrow{c} \ell_3 \\
\end{align*}
\]

<table>
<thead>
<tr>
<th>State</th>
<th>(y=0)</th>
<th>x=2, c</th>
</tr>
</thead>
<tbody>
<tr>
<td>\ell_0</td>
<td>+5</td>
<td>+10</td>
</tr>
<tr>
<td>\ell_1</td>
<td>1.3</td>
<td>u</td>
</tr>
<tr>
<td>\ell_2</td>
<td>0</td>
<td>c</td>
</tr>
<tr>
<td>\ell_3</td>
<td>2</td>
<td>0.7</td>
</tr>
</tbody>
</table>

cost :

\[
\ell_0 \xrightarrow{1.3} \ell_0 \xrightarrow{c} \ell_1 \xrightarrow{u} \ell_3 \xrightarrow{0.7} \ell_3 \xrightarrow{c} /\mathsf{smiley}
\]

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata (HSCC'01).

Weighted/priced timed automata [ALP01,BFH+01]

\[
\ell_0 \xrightarrow{+5} \ell_0 \\
\ell_0 \xrightarrow{x \leq 2, c, y := 0} \ell_1 \\
\ell_1 \xrightarrow{(y = 0)} \ell_2 \\
\ell_2 \xrightarrow{+10} \ell_2 \\
\ell_2 \xrightarrow{x = 2, c} \ell_3 \\
\ell_3 \xrightarrow{+1} \ell_3 \\
\ell_3 \xrightarrow{x = 2, c} \ell_0 \\
\ell_3 \xrightarrow{c} \ell_3 \\
\ell_3 \xrightarrow{c} \smiley
\]

<table>
<thead>
<tr>
<th>State</th>
<th>( \ell_0 )</th>
<th>( \ell_1 )</th>
<th>( \ell_2 )</th>
<th>( \ell_3 )</th>
<th>( \ell_3 )</th>
<th>( \ell_3 )</th>
<th>( \ell_0 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x )</td>
<td>0</td>
<td>1.3</td>
<td>1.3</td>
<td>1.3</td>
<td>2</td>
<td>0.7</td>
<td>0.7</td>
</tr>
<tr>
<td>( y )</td>
<td>0</td>
<td>1.3</td>
<td>0</td>
<td>0</td>
<td>0.7</td>
<td>0.7</td>
<td>0.7</td>
</tr>
</tbody>
</table>

\[
\text{cost: } 6.5
\]

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata (HSCC'01).

Weighted/priced timed automata [ALP01,BFH+01]

\[
\ell_0 + 5 \xrightarrow{x \leq 2, c, y := 0} \ell_1 \xrightarrow{(y = 0)} \ell_2 \xrightarrow{u} \ell_3 + 1 \xrightarrow{x = 2, c} \ell_2 + 10 \xrightarrow{u} \ell_1 \xrightarrow{x = 2, c} \ell_3 + 1
\]

\[
\begin{array}{cccccc}
\ell_0 & 1.3 & \ell_0 & c & \ell_1 & u & \ell_3 & 0.7 & \ell_3 & c & \text{smiley} \\
\hline
x & 0 & 1.3 & 1.3 & 1.3 & 2 \\
y & 0 & 1.3 & 0 & 0 & 0.7
\end{array}
\]

cost : 6.5 + 0


Weighted/priced timed automata [ALP01,BFH+01]

\[
\begin{align*}
\ell_0 & \xrightarrow{1.3} \ell_1 \\
\ell_1 & \xrightarrow{c} \ell_1 \\
\ell_1 & \xrightarrow{u} \ell_3 \\
\ell_3 & \xrightarrow{0.7} \ell_3 \\
\ell_3 & \xrightarrow{c} \text{smiley}
\end{align*}
\]

\begin{align*}
\ell_0 & \xrightarrow{5} \ell_1 \\
\ell_1 & \xrightarrow{c} \ell_1 \\
\ell_1 & \xrightarrow{u} \ell_3 \\
\ell_3 & \xrightarrow{+1} \\
\ell_3 & \xrightarrow{+1} \\
\ell_3 & \xrightarrow{+10} \\
\ell_3 & \xrightarrow{x=2,c} \text{smiley}
\end{align*}

\[
\begin{align*}
x & \leq 2, c, y := 0 \\
(y=0)
\end{align*}
\]

\begin{align*}
\ell_0 & \xrightarrow{x=2,c} \ell_0 \\
\ell_0 & \xrightarrow{c} \ell_1 \\
\ell_1 & \xrightarrow{u} \ell_3 \\
\ell_3 & \xrightarrow{c} \text{smiley}
\end{align*}

\[
\begin{align*}
x & = 0, 1.3, 1.3, 1.3, 1.3, 1.3, 2 \\
y & = 0, 1.3, 0, 0, 0, 0, 0.7
\end{align*}
\]

cost : \[6.5 + 0 + 0 \]

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata (HSCC'01).

Weighted/priced timed automata [ALP01,BFH+01]

\[
\begin{align*}
\ell_0 & \xrightarrow{x \leq 2, c, y := 0} \ell_1 \\
& \xrightarrow{u} \ell_2 \\
& \xrightarrow{x = 2, c} \ell_3 \\
& \xrightarrow{c} \text{smiley}
\end{align*}
\]

\[
\begin{array}{c|cc|cc|cc|c}
\ell_0 & 1.3 & \ell_0 & c & \ell_1 & u & \ell_3 & 0.7 & c \\
x & 0 & 1.3 & 1.3 & 1.3 & 2 & 0.7 \\
y & 0 & 1.3 & 0 & 0 & 0.7 \\
\text{cost :} & 6.5 & + & 0 & + & 0 & + & 0.7
\end{array}
\]


Modelling and optimizing resources in timed systems

Weighted/priced timed automata [ALP01,BFH+01]

\[ \ell_0 \xrightarrow{x \leq 2,c,y:=0} \ell_1 \xrightarrow{u} \ell_2 \xrightarrow{x=2,c} \ell_3 \xrightarrow{u} \ell_3 \xrightarrow{x=2,c} \ell_3 \xrightarrow{c} \text{smiley} \]

\[ \ell_0 \xrightarrow{1.3} \ell_0 \xrightarrow{c} \ell_1 \xrightarrow{u} \ell_3 \xrightarrow{0.7} \ell_3 \xrightarrow{c} \text{smiley} \]

<table>
<thead>
<tr>
<th>State</th>
<th>( \ell_0 )</th>
<th>( \ell_0 )</th>
<th>( c )</th>
<th>( \ell_1 )</th>
<th>( u )</th>
<th>( \ell_3 )</th>
<th>( c )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x )</td>
<td>0</td>
<td>1.3</td>
<td>1.3</td>
<td>1.3</td>
<td>1.3</td>
<td>2</td>
<td>0.7</td>
</tr>
<tr>
<td>( y )</td>
<td>0</td>
<td>1.3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.7</td>
<td>7</td>
</tr>
</tbody>
</table>

Cost: \( 6.5 + 0 + 0 + 0.7 + 7 \)
Weighted/priced timed automata [ALP01,BFH+01]

![Diagram of a weighted/priced timed automaton](image)

\[
\begin{array}{c}
\ell_0 & \xrightarrow{+5} & \ell_0 \\
& & x \leq 2, c, y := 0 \\
\ell_0 & \xrightarrow{1.3} & \ell_0 \\
& & x = 2, c, y := 0 \\
\ell_1 & \xrightarrow{(y=0)} & \ell_1 \\
& & x = 2, c \\
\ell_2 & \xrightarrow{u} & \ell_2 \\
& & x = 2, c \\
\ell_3 & \xrightarrow{u} & \ell_3 \\
& & x = 2, c \\
\end{array}
\]

<table>
<thead>
<tr>
<th>State</th>
<th>( x )</th>
<th>( y )</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \ell_0 )</td>
<td>0.00</td>
<td>0.00</td>
<td>6.50</td>
</tr>
<tr>
<td>( \ell_1 )</td>
<td>1.30</td>
<td>0.00</td>
<td>6.80</td>
</tr>
<tr>
<td>( \ell_2 )</td>
<td>1.30</td>
<td>0.00</td>
<td>7.10</td>
</tr>
<tr>
<td>( \ell_3 )</td>
<td>2.00</td>
<td>0.70</td>
<td>2.70</td>
</tr>
<tr>
<td>( \text{Smiley} )</td>
<td>7.00</td>
<td>0.00</td>
<td>7.00</td>
</tr>
</tbody>
</table>

Cost: 6.50 + 0.00 + 0.70 + 7.00 = 14.20

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata (HSCC'01).
Weighted/priced timed automata \[\text{[ALP01,BFH+01]}\]

Question: what is the optimal cost for reaching \(\text{😊}\)?
Weighted/priced timed automata [ALP01,BFH+01]

Question: what is the optimal cost for reaching 😊?

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

Weighted/priced timed automata \cite{ALP01,BFH+01}

Question: what is the optimal cost for reaching \smiley

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

\[ \inf_{0 \leq t \leq 2} \min \]
Question: what is the optimal cost for reaching 😊?

$$\min \left( 5t + 10(2 - t) + 1, \ 5t + (2 - t) + 7 \right)$$

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata (HSCC'01).
Weighted/priced timed automata [ALP01,BFH+01]

Question: what is the optimal cost for reaching 🌞?  

\[
\inf_{0 \leq t \leq 2} \min \left( 5t + 10(2 - t) + 1, 5t + (2 - t) + 7 \right) = 9
\]
Weighted/priced timed automata [ALP01,BFH+01]

\[\ell_0 + 5 \xrightarrow{x \leq 2, c, y := 0} \ell_1 \xrightarrow{} \ell_2 \xrightarrow{x = 2, c} \ell_3 \xrightarrow{+1} \text{smiley} \]

**Question:** what is the optimal cost for reaching \(\text{smiley} \)?

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

\(\leadsto\) **strategy:** leave immediately \(\ell_0\), go to \(\ell_3\), and wait there 2 t.u.


The region abstraction is not fine enough

[Diagram showing transition from one region to another with arrows indicating time elapsing and reset to 0]
The corner-point abstraction

We can somehow discretize the behaviours...
The corner-point abstraction

We can somehow discretize the behaviours...
From timed to discrete behaviours

Optimal reachability as a linear programming problem
From timed to discrete behaviours

Optimal reachability as a linear programming problem

\[
\begin{align*}
\text{t}_1 & \rightarrow \text{t}_2 & \rightarrow \text{t}_3 & \rightarrow \text{t}_4 & \rightarrow \text{t}_5 \rightarrow \ldots
\end{align*}
\]
From timed to discrete behaviours

Optimal reachability as a linear programming problem

\[ t_1 + t_2 \leq 2 \]
From timed to discrete behaviours

Optimal reachability as a linear programming problem

\[ \begin{align*}
  t_1 & \quad t_2 \quad t_3 \quad t_4 \quad t_5 \quad \ldots \\
  y := 0 & \quad x \leq 2 & \quad y \geq 5 & \quad t_1 + t_2 \leq 2 \\
  t_2 + t_3 + t_4 & \geq 5
\end{align*} \]
From timed to discrete behaviours

**Optimal reachability as a linear programming problem**

![Diagram with timed transitions and inequalities]

**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.
From timed to discrete behaviours

Optimal reachability as a linear programming problem

\[ \begin{align*}
  t_1 & \rightarrow t_2 \rightarrow t_3 \rightarrow t_4 \rightarrow t_5 \rightarrow \ldots \\
  y & := 0 \quad x \leq 2 \quad y \geq 5
\end{align*} \]

\[ \begin{align*}
  t_1 + t_2 & \leq 2 \\
  t_2 + t_3 + t_4 & \geq 5
\end{align*} \]

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.

\( \rightarrow \) for every finite path \( \pi \) in \( A \), there exists a path \( \Pi \) in \( A_{cp} \) such that

\[ \text{cost}(\Pi) \leq \text{cost}(\pi) \]

[\( \Pi \) is a “corner-point projection” of \( \pi \)]
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.

$$\|\Pi - \pi_\varepsilon\|_\infty < \varepsilon$$
Approximation of abstract paths:

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

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

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

$$\|\Pi - \pi_\varepsilon\|_\infty < \varepsilon \Rightarrow |\text{cost}(\Pi) - \text{cost}(\pi_\varepsilon)| < \eta$$
Optimal-cost reachability

**Theorem [ALP01,BFH+01,BBBR07]**

The optimal-cost reachability problem is decidable (and PSPACE-complete) in (weighted) timed automata.

---


Going further

The corner-point abstraction can be used for the following problems:

- mean-cost optimization problem [BBL04, BBL08]
- discounted cost optimization problem [FL08]
- concavely-priced cost optimization problem [JT08]
A fourth model of the system
What if there is an unexpected event?
A fourth model of the system
What if there is an unexpected event?
A fourth model of the system

What if there is an unexpected event?

Modelling and optimizing resources in timed systems

∽ modelled as timed games
A simple example of timed game

\[ x \leq 2, c, y := 0 \]
A simple example of timed game

\[ \ell_0 \xrightarrow{x \leq 2, c, y := 0} \ell_1 \xrightarrow{(y = 0)} \ell_2 \xrightarrow{x = 2, c} \ell_3 \xrightarrow{x = 2, c} \text{Smiley} \]
Another example

\[ \ell_0 \quad \ell_1 \quad \ell_2 \quad \ell_3 \]

\( x \leq 2 \)
\( x \geq 1 \)
\( x \leq 1 \)
\( x < 1, x := 0 \)
\( x < 1 \)

\( x \geq 2 \)
\( x \leq 1 \)
Decidability of timed games

Theorem [AMPS98,HK99] Safety and reachability control in timed automata are decidable and EXPTIME-complete.

Decidability of timed games

**Theorem [AMPS98, HK99]**

Safety and reachability control in timed automata are decidable and EXPTIME-complete.

(the attractor is computable...)


Decidability of timed games

**Theorem [AMPS98,HK99]**

Safety and reachability control in timed automata are decidable and EXPTIME-complete.

(the attractor is computable...)

$\leadsto$ classical regions are sufficient for solving such problems
Decidability of timed games

**Theorem [AMPS98,HK99]**

Safety and reachability control in timed automata are decidable and EXPTIME-complete.

(the attractor is computable...)

\[ \leadsto \text{classical regions are sufficient for solving such problems} \]

**Theorem [AM99,BHPR07,JT07]**

Optimal-time reachability timed games are decidable and EXPTIME-complete.

---

[AM99] Asarin, Maler. As soon as possible: time optimal control for timed automata (*HSCC’99*).


[JT07] Jurdziński, Trivedi. Reachability-time games on timed automata (*ICALP’07*).
Back to the simple example

\[ x \leq 2, c, y := 0 \]

\[ (y = 0) \]

Question: what is the optimal cost we can ensure while reaching

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

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

\[ \ell_2 \]

\[ x = 2, c \]

\[ \ell_3 \]

\[ x = 2, c \]
Back to the simple example

$\ell_0 \xrightarrow{x \leq 2, c, y := 0} \ell_1 \xrightarrow{(y = 0)} \ell_3 \xrightarrow{x = 2, c} \ell_2 \xrightarrow{x = 2, c} \text{smiley}$

Question: what is the optimal cost we can ensure while reaching the smiley?

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

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

(x=2, c)
Back to the simple example

Question: what is the optimal cost we can ensure while reaching 😊?
Back to the simple example

Question: what is the optimal cost we can ensure while reaching 😊?

\[ 5t + 10(2 - t) + 1 \]
Back to the simple example

\[ \ell_0 + 5 \xrightarrow{x \leq 2, c, y := 0} \ell_1 \xrightarrow{(y = 0)} \ell_2 \xrightarrow{+10} \ell_3 \xrightarrow{x = 2, c} +1 \]

**Question:** what is the optimal cost we can ensure while reaching \( \bigcirc \)?

\[ 5t + 10(2 - t) + 1, \quad 5t + (2 - t) + 7 \]
Back to the simple example

Question: what is the optimal cost we can ensure while reaching \( \smiley \)?

\[
\max \left( 5t + 10(2 - t) + 1 , \ 5t + (2 - t) + 7 \right)
\]
Back to the simple example

\[
\ell_0 + 5 \xrightarrow{x \leq 2, c, y := 0} \ell_1 \quad \text{(y = 0)} \quad \ell_2 + 10 \xrightarrow{x = 2, c} \ell_3 + 1 \quad \smiley
\]

**Question:** what is the optimal cost we can ensure while reaching \( \smiley \)?

\[
\inf_{0 \leq t \leq 2} \max \left( 5t + 10(2 - t) + 1, \ 5t + (2 - t) + 7 \right) = 14 + \frac{1}{3}
\]
Question: what is the optimal cost we can ensure while reaching 😊?

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

\[\leadsto \text{strategy: wait in } \ell_0, \text{ and when } t = \frac{4}{3}, \text{ go to } \ell_1\]
Optimal reachability in weighted timed games

This topic has been fairly hot these last couple of years...

e.g. [LMM02,ABM04,BCFL04]

---

[LMM02] La Torre, Mukhopadhyay, Murano. Optimal-reachability and control for acyclic weighted timed automata (TCS@02).
Optimal reachability in weighted timed games

This topic has been fairly hot these last couple of years...

e.g. [LMM02,ABM04,BCFL04]

**Theorem [BBR05,BBM06]**

Optimal timed games are **undecidable**, as soon as automata have three clocks or more.

[BBR05] Brihaye, Bruyère, Raskin. On optimal timed strategies (*FORMATS’05*).

[BBM06] Bouyer, Brihaye, Markey. Improved undecidability results on weighted timed automata (*Information Processing Letters*).
Optimal reachability in weighted timed games

This topic has been fairly hot these last couple of years...

e.g. [LMM02,ABM04,BCFL04]

Theorem [BBR05,BBM06]
Optimal timed games are undecidable, as soon as automata have three clocks or more.

Theorem [BLMR06]
Turn-based optimal timed games are decidable in 3EXPTIME when automata have a single clock. They are PTIME-hard.

[BBR05] Brihaye, Bruyère, Raskin. On optimal timed strategies (FORMATS’05).
[BBM06] Bouyer, Brihaye, Markey. Improved undecidability results on weighted timed automata (Information Processing Letters).
The positive side

Theorem [BLMR06]

Turn-based optimal timed games are decidable in 3EXPTIME when automata have a single clock. They are PTIME-hard.

- Key: resetting the clock somehow resets the history...

The positive side

**Theorem [BLMR06]**

Turn-based optimal timed games are **decidable in 3EXPTIME** when automata have a single clock. They are **PTIME-hard**.

- Key: resetting the clock somehow resets the history...
- Memoryless strategies can be non-optimal...

---

(BLMR06) Bouyer, Larsen, Markey, Rasmussen. Almost-optimal strategies in one-clock priced timed automata (*FSTTCS’06*).
The positive side

**Theorem [BLMR06]**

Turn-based optimal timed games are decidable in \(3\text{EXPTIME}\) when automata have a single clock. They are PTIME-hard.

- Key: resetting the clock somehow resets the history...
- Memoryless strategies can be non-optimal...

\[
\begin{align*}
\ell_0 & \quad \text{when } x \leq 1 \\
\ell_1 & \quad \text{when } x < 1, x := 0 \\
\ell_1 & \quad \text{when } x > 0
\end{align*}
\]

- However, by unfolding and removing one by one the locations, we can synthesize memoryless almost-optimal winning strategies.

The positive side

**Theorem [BLMR06]**

Turn-based optimal timed games are decidable in $3\text{EXPTIME}$ when automata have a single clock. They are PTIME-hard.

- Key: resetting the clock somehow resets the history...
- Memoryless strategies can be non-optimal...

![Diagram](attachment:image_url)

- However, by unfolding and removing one by one the locations, we can synthesize memoryless almost-optimal winning strategies.
- Rather involved proof of correctness for a simple algorithm.

The negative side: why is that hard?

Given two clocks $x$ and $y$, we can check whether $y = 2x$. 
The negative side: why is that hard?

Given two clocks $x$ and $y$, we can check whether $y = 2x$.

The cost is increased by $x_0$

The cost is increased by $1 - x_0$
The negative side: why is that hard?

Given two clocks $x$ and $y$, we can check whether $y = 2x$. 

In , cost = 2$x$ + (1 − $y$) + 2

In , cost = 2(1 − $x$) + $y$ + 1

If $y < 2x$, player 2 chooses the first branch: cost > 3

If $y > 2x$, player 2 chooses the second branch: cost > 3

If $y = 2x$, in both branches, cost = 3

Player 1 has a winning strategy with cost $\leq 3$ iff $y = 2x$. 
The negative side: why is that hard?

Given two clocks $x$ and $y$, we can check whether $y = 2x$.

\[
\begin{align*}
\text{If } y_0 < 2x_0, \text{ player 2 chooses the first branch: cost } &> 3 \\
\text{If } y_0 > 2x_0, \text{ player 2 chooses the second branch: cost } &> 3 \\
\text{If } y_0 = 2x_0, \text{ in both branches, cost } &= 3
\end{align*}
\]

\[\text{In } \begin{cases} \text{ if } y_0 < 2x_0, \text{ cost } = 2x_0 + (1 - y_0) + 2 \\
\text{ if } y_0 > 2x_0, \text{ cost } = 2x_0 + (1 - y_0) + 2 \\
\text{ if } y_0 = 2x_0, \text{ cost } = 3 \end{cases} \]
The negative side: why is that hard?

Given two clocks $x$ and $y$, we can check whether $y = 2x$.

In [green], cost = $2x_0 + (1 - y_0) + 2$

In [purple], cost = $2(1 - x_0) + y_0 + 1$
The negative side: why is that hard?

Given two clocks $x$ and $y$, we can check whether $y = 2x$.

\[
\begin{align*}
\text{In } \bigcirc, & \quad \text{cost} = 2x_0 + (1 - y_0) + 2 \\
\text{In } \bigcirc, & \quad \text{cost} = 2(1 - x_0) + y_0 + 1 \\
\text{if } y_0 < 2x_0, & \quad \text{player 2 chooses the first branch: cost > 3}
\end{align*}
\]
The negative side: why is that hard?

Given two clocks \( x \) and \( y \), we can check whether \( y = 2x \).

\[
\begin{align*}
\text{if } y_0 < 2x_0, & \quad \text{player 2 chooses the first branch: cost } > 3 \\
\text{if } y_0 > 2x_0, & \quad \text{player 2 chooses the second branch: cost } > 3
\end{align*}
\]
The negative side: why is that hard?

Given two clocks $x$ and $y$, we can check whether $y = 2x$.

- In $\bigcirc$, cost = $2x_0 + (1 - y_0) + 2$
- In $\bigotimes$, cost = $2(1 - x_0) + y_0 + 1$

- if $y_0 < 2x_0$, player 2 chooses the first branch: cost $> 3$
- if $y_0 > 2x_0$, player 2 chooses the second branch: cost $> 3$
- if $y_0 = 2x_0$, in both branches, cost = 3
The negative side: why is that hard?

Given two clocks $x$ and $y$, we can check whether $y = 2x$.

In $\bigcirc$, cost $= 2x_0 + (1 - y_0) + 2$

In $\bigstar$, cost $= 2(1 - x_0) + y_0 + 1$

- If $y_0 < 2x_0$, player 2 chooses the first branch: cost $> 3$
- If $y_0 > 2x_0$, player 2 chooses the second branch: cost $> 3$
- If $y_0 = 2x_0$, in both branches, cost $= 3$

Player 1 has a winning strategy with cost $\leq 3$ iff $y_0 = 2x_0$
The negative side: why is that hard?

Player 1 will simulate a two-counter machine:

- each instruction is encoded as a module;
- the values $c_1$ and $c_2$ of the counters are encoded by the values of two clocks:

$$x = \frac{1}{2c_1} \quad \text{and} \quad y = \frac{1}{3c_2}$$

when entering the corresponding module.
The negative side: why is that hard?

Player 1 will simulate a two-counter machine:
- each instruction is encoded as a module;
- the values $c_1$ and $c_2$ of the counters are encoded by the values of two clocks:

$$x = \frac{1}{2c_1} \quad \text{and} \quad y = \frac{1}{3c_2}$$

when entering the corresponding module.

The two-counter machine has an halting computation iff player 1 has a winning strategy to ensure a cost no more than 3.
The negative side: why is that hard?

Player 1 will simulate a two-counter machine:
- each instruction is encoded as a module;
- the values $c_1$ and $c_2$ of the counters are encoded by the values of two clocks:
  \[ x = \frac{1}{2c_1} \quad \text{and} \quad y = \frac{1}{3c_2} \]
  when entering the corresponding module.

The two-counter machine has an halting computation iff player 1 has a winning strategy to ensure a cost no more than 3.

Globally, $(x \leq 1, y \leq 1, u \leq 1)$

\[
\begin{align*}
  &x=1, x:=0 \quad \text{or} \quad y=1, y:=0 \\
  &u:=0 \\
\end{align*}
\]

The diagram shows the transition states with the following conditions:

- $x = \frac{1}{2c}$
- $y = \frac{1}{2d}$
- $z = \star$

- $x = \frac{1}{2c}$
- $y = \frac{1}{2d}$
- $z = \alpha$

Test $y(x=2z)$
Outline

1. Introduction

2. Modelling and optimizing resources in timed systems

3. Managing resources

4. Conclusion
A fifth model of the system
Can I work with my computer all the way?
Can I work with my computer all the way?
Can I work with my computer all the way?

Energy is not only consumed, but can be regained. 
\[ \leadsto \] the aim is to continuously satisfy some energy constraints.
An example of resource management

Globally \((x \leq 1)\)

\[
\ell_0 \xrightarrow{-3} \ell_1 \xrightarrow{+6} \ell_2 \xrightarrow{-6} \ell_0
\]

\(x := 0 \quad x = 1\)
An example of resource management

Globally \((x \leq 1)\)

-3 \(\ell_0\) → +6 \(\ell_1\) → −6 \(\ell_2\)

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

- Lower-bound problem: can we stay above 0?
An example of resource management

Globally \((x \leq 1)\)

Lower-bound problem: can we stay above 0?
Managing resources

A new approach to managing resources

An example of resource management

Globally ($x \leq 1$)

Lower-bound problem: can we stay above 0?

Lower-weak-upper-bound problem: can we "weakly" stay within bounds?
An example of resource management

Globally \((x \leq 1)\)

\[
\begin{align*}
-3 & \quad \ell_0 \\
+6 & \quad \ell_1 \\
-6 & \quad \ell_2
\end{align*}
\]

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

\bullet \text{ Lower-bound problem: can we stay above 0?}
An example of resource management

Globally \((x \leq 1)\)

\[
\ell_0 - 3 \rightarrow \ell_1 + 6 \rightarrow \ell_2 - 6
\]

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

- Lower-bound problem: can we stay above 0?
An example of resource management

Globally $(x \leq 1)$

$-3 \rightarrow +6 \rightarrow -6$

$x:=0 \quad x=1$

- Lower-bound problem: can we stay above 0?
An example of resource management

Globally \((x \leq 1)\)

\[
-3 \ell_0 \rightarrow +6 \ell_1 \rightarrow -6 \ell_2
\]

\(x := 0\) \hspace{1cm} x = 1

- Lower-bound problem
- Lower-upper-bound problem: can we stay within bounds?
An example of resource management

Globally \((x \leq 1)\)

Lower-bound problem

Lower-upper-bound problem: \textit{can we stay within bounds?}
An example of resource management

Globally \((x \leq 1)\)

\[
\begin{align*}
-3 \ell_0 & \quad +6 \ell_1 & \quad -6 \ell_2 \\
\downarrow x:=0 & \quad \quad & \quad \quad x=1
\end{align*}
\]

- Lower-bound problem
- Lower-upper-bound problem: can we stay within bounds?
An example of resource management

Globally ($x \leq 1$)

Lower-bound problem

Lower-upper-bound problem: can we stay within bounds?
An example of resource management

Globally \((x \leq 1)\)

-3 \(\rightarrow\) +6 \(\rightarrow\) -6

\(\ell_0 \rightarrow \ell_1 \rightarrow \ell_2\)

\(x := 0\)

\(x := 1\)

- Lower-bound problem
- Lower-upper-bound problem: can we stay within bounds?
An example of resource management

Globally \((x \leq 1)\)

\[
\ell_0 - 3 \rightarrow \ell_1 + 6 \rightarrow \ell_2 - 6
\]

\(x := 0\) \quad \text{for} \quad x = 1

- Lower-bound problem
- Lower-upper-bound problem: can we stay within bounds?
An example of resource management

Globally \((x \leq 1)\)

Lower-bound problem

Lower-upper-bound problem: can we stay within bounds?
An example of resource management

Globally \((x \leq 1)\)

- Lower-bound problem
- Lower-upper-bound problem: can we stay within bounds?

lost!
An example of resource management

Globally \((x \leq 1)\)

-3 \(\ell_0\) \(\rightarrow\) +6 \(\ell_1\) \(\rightarrow\) -6 \(\ell_2\)

\(x := 0\) \(\rightarrow\) \(x = 1\)

- Lower-bound problem
- Lower-upper-bound problem: can we stay within bounds?
An example of resource management

Globally ($x \leq 1$)

Lower-bound problem

Lower-upper-bound problem: can we stay within bounds?
An example of resource management

Globally \((x \leq 1)\)

\[-3 \ell_0 + 6 \ell_1 - 6 \ell_2 = 1\]

\(x := 0\) \hspace{1cm} \(x = 1\)

- Lower-bound problem
- Lower-upper-bound problem: can we stay within bounds?

lost!
An example of resource management

Globally \((x \leq 1)\)

\[
\begin{align*}
\ell_0 &\rightarrow \ell_1 \rightarrow \ell_2 \\
x := 0 &\rightarrow x = 1
\end{align*}
\]

- Lower-bound problem
- Lower-upper-bound problem
- **Lower-weak-upper-bound problem**: can we “weakly” stay within bounds?
An example of resource management

Globally ($x \leq 1$)

- $\ell_0$ to $\ell_1$: $-3 + 6$
- $\ell_1$ to $\ell_2$: $-6$

$x := 0$ to $x = 1$

- Lower-bound problem $\leadsto L$
- Lower-upper-bound problem $\leadsto L + U$
- Lower-weak-upper-bound problem $\leadsto L + W$
Only partial results so far [BFLMS08]

<table>
<thead>
<tr>
<th>0 clock!</th>
<th>exist. problem</th>
<th>univ. problem</th>
<th>games</th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
<td>∈ PTIME</td>
<td>∈ PTIME</td>
<td>∈ UP ∩ co-UP PTIME-hard</td>
</tr>
<tr>
<td>L+W</td>
<td>∈ PTIME</td>
<td>∈ PTIME</td>
<td>∈ NP ∩ co-NP PTIME-hard</td>
</tr>
<tr>
<td>L+U</td>
<td>∈ PSPACE NP-hard</td>
<td>∈ PTIME</td>
<td>EXPTIME-c.</td>
</tr>
</tbody>
</table>
Only partial results so far [BFLMS08]

<table>
<thead>
<tr>
<th>1 clock</th>
<th>exist. problem</th>
<th>univ. problem</th>
<th>games</th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
<td>$\in$ PTIME</td>
<td>$\in$ PTIME</td>
<td>?</td>
</tr>
<tr>
<td>L+W</td>
<td>$\in$ PTIME</td>
<td>$\in$ PTIME</td>
<td>?</td>
</tr>
<tr>
<td>L+U</td>
<td>?</td>
<td>?</td>
<td>undecidable</td>
</tr>
</tbody>
</table>

Only partial results so far \cite{Bouyer:2008a}.

<table>
<thead>
<tr>
<th>$n$ clocks</th>
<th>exist. problem</th>
<th>univ. problem</th>
<th>games</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L$</td>
<td>$?$</td>
<td>$?$</td>
<td>$?$</td>
</tr>
<tr>
<td>$L+W$</td>
<td>$?$</td>
<td>$?$</td>
<td>$?$</td>
</tr>
<tr>
<td>$L+U$</td>
<td>$?$</td>
<td>$?$</td>
<td>undecidable</td>
</tr>
</tbody>
</table>

\cite{Bouyer:2008a} Bouyer, Fahrenberg, Larsen, Markey, Srba. Infinite runs in weighted timed automata with energy constraints (FORMATS'08).
Relation with mean-payoff games

**Definition**

Mean-payoff games: in a weighted game graph, does there exists a strategy s.t. the mean-cost of any play is nonnegative?
Relation with mean-payoff games

<table>
<thead>
<tr>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mean-payoff games:</strong> in a weighted game graph, does there exists a strategy s.t. the mean-cost of any play is nonnegative?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Lemma</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>$L$-games and $L+W$-games</strong> are determined, and memoryless strategies are sufficient to win.</td>
</tr>
</tbody>
</table>
Relation with mean-payoff games

**Definition**
Mean-payoff games: in a weighted game graph, does there exists a strategy s.t. the mean-cost of any play is nonnegative?

**Lemma**
$L$-games and $L + W$-games are determined, and memoryless strategies are sufficient to win.

- from mean-payoff games to $L$-games or $L + W$-games: play in the same game graph $G$ with initial credit $-M \geq 0$ (where $M$ is the sum of negative costs in $G$).
Relation with mean-payoff games

Definition

Mean-payoff games: in a weighted game graph, does there exists a strategy s.t. the mean-cost of any play is nonnegative?

Lemma

$L$-games and $L+W$-games are determined, and memoryless strategies are sufficient to win.

- from mean-payoff games to $L$-games or $L+W$-games: play in the same game graph $G$ with initial credit $-M \geq 0$ (where $M$ is the sum of negative costs in $G$).
- from $L$-games to mean-payoff games: transform the game as follows:

```
  p
  □\rightarrow 0
  \rightarrow 0
```
to initial state
The single-clock $L+U$-games are undecidable.

We encode the behavior of a two-counter machine: each instruction is encoded as a module; the values $c_1$ and $c_2$ of the counters are encoded by the energy level $e = 5 - 2^{c_1} \cdot 3^{c_2}$ when entering the corresponding module.

There is an infinite execution in the two-counter machine iff there is a strategy in the single-clock timed game under which the energy level remains between 0 and 5.

We present a generic construction for incrementing/decrementing the counters.
The single-clock $L+U$-games are undecidable.

We encode the behaviour of a two-counter machine:

- each instruction is encoded as a module;
- the values $c_1$ and $c_2$ of the counters are encoded by the energy level

$$e = 5 - \frac{1}{2^{c_1} \cdot 3^{c_2}}$$

when entering the corresponding module.
Single-clock $\mathbf{L+U}$-games

Theorem
The single-clock $\mathbf{L+U}$-games are undecidable.

We encode the behaviour of a two-counter machine:
- each instruction is encoded as a module;
- the values $c_1$ and $c_2$ of the counters are encoded by the energy level

$$e = 5 - \frac{1}{2^{c_1} \cdot 3^{c_2}}$$

when entering the corresponding module.

There is an infinite execution in the two-counter machine iff there is a strategy in the single-clock timed game under which the energy level remains between 0 and 5.
Single-clock $L+U$-games

Theorem
The single-clock $L+U$-games are undecidable.

We encode the behaviour of a two-counter machine:
- each instruction is encoded as a module;
- the values $c_1$ and $c_2$ of the counters are encoded by the energy level

\[ e = 5 - \frac{1}{2^{c_1} \cdot 3^{c_2}} \]

when entering the corresponding module.

There is an infinite execution in the two-counter machine iff there is a strategy in the single-clock timed game under which the energy level remains between 0 and 5.

$\leadsto$ We present a generic construction for incrementing/decrementing the counters.
Generic module for incrementing/decrementing

$x:=0$

$m$

$-6$

$m_1$

$-6$

$m_2$

$+30$

$m_3$

$+30$

$n$

$-\alpha$

$x:=0$

$x:=0$

$x:=0$

$x:=0$

$\alpha=3$: increment

$\alpha=2$: increment

$\alpha=12$: decrement

$\alpha=18$: decrement

module ok

module ok
Generic module for incrementing/decrementing

\[ x := 0 \quad m \quad x := 0 \quad m_1 \quad x := 1 \quad m_2 \quad x := 1 \quad m_3 \quad x := 1 \quad n \]

\[ m_2 + 30 \quad m_3 + 30 \quad n \]

\[ x := 0 \quad m \quad x := 0 \quad m_1 \quad x := 1 \quad m_2 \quad x := 1 \quad m_3 \quad x := 1 \quad n \]

\[ \text{module ok} \quad \text{module ok} \]

\[ \alpha \]

\[ x = 0 \]

\[ 5 - e \]

\[ 0 \quad 1 \]

\[ x \]

\[ \text{energy} \]
Generic module for incrementing/decrementing

Managing resources

\[ m - 6 \rightarrow m_1 - 6 \rightarrow m_2 + 30 \rightarrow m_3 + 30 \rightarrow n - \alpha \]

\[ x := 0 \rightarrow \text{module } \textbf{ok} \rightarrow x := 0 \rightarrow \text{module } \textbf{ok} \]

\[
\begin{align*}
5 - e & \quad \text{energy} \\
0 & \quad \text{x} \\
1 & \quad \text{x}
\end{align*}
\]
Managing resources

Generic module for incrementing/decrementing

\[ x := 0 \rightarrow -6 \rightarrow -6 \rightarrow +30 \rightarrow +30 \rightarrow -\alpha \rightarrow x = 1 \]

\[ m \rightarrow m_1 \rightarrow m_2 \rightarrow m_3 \rightarrow n \]

\[ x := 0 \rightarrow +5 \rightarrow x = 1 \rightarrow \text{module ok} \]

\[ x := 0 \rightarrow -5 \rightarrow x = 1 \rightarrow \text{module ok} \]

\[ \alpha \]

\[ x := 0 \rightarrow 5 - e \rightarrow x = 1 \rightarrow \frac{5 - e}{6} \rightarrow 1 \]

energy
Generic module for incrementing/decrementing

\[ m \rightarrow m_1 \rightarrow m_2 \rightarrow m_3 \rightarrow n \]

\[ x := 0 \]

\[ +5 \]

\[ x := 0 \]

\[ x = 1 \]

Module ok

\[ -\alpha \]

\[ x = 1 \]

Module ok

Energy graph:

\[ 5 - e \]

\[ x \]

\[ 0 \]

\[ \frac{5 - e}{6} \]

\[ 1 \]
Generic module for incrementing/decrementing

Managing resources
Generic module for incrementing/decrementing

\[ x := 0 \]

\[ m \]

\[ m_1 \]

\[ m_2 \]

\[ m_3 \]

\[ n \]

\[ x = 1 \]

\[ x := 0 \]

\[ x = 0 \]

\[ x = 1 \]

\[ x = 1 \]

\[ \text{module ok} \]

\[ \text{module ok} \]

\[ \text{energy} \]

\[ 5 - e \]

\[ 5 - \frac{\alpha e}{6} \]

\[ 0 \]

\[ x \]

\[ \frac{5 - e}{6} \]
Managing resources

Generic module for incrementing/decrementing

\[
x := 0 \\
m \rightarrow -6 \rightarrow m_1 \rightarrow +30 \rightarrow m_2 \rightarrow +30 \rightarrow m_3 \rightarrow - \alpha \rightarrow n \\
x := 0 \\
x := 1
\]

\[x := 0\] to \[x := 1\]

\[
\begin{align*}
\text{module ok} & \quad \text{module ok}
\end{align*}
\]

energy

\[
\begin{align*}
\alpha = 3: & \quad \text{increment } c_1 \\
\alpha = 2: & \quad \text{increment } c_2 \\
\alpha = 12: & \quad \text{decrement } c_1 \\
\alpha = 18: & \quad \text{decrement } c_2
\end{align*}
\]
Outline

1. Introduction
2. Modelling and optimizing resources in timed systems
3. Managing resources
4. Conclusion
Conclusion

- **Priced/weighted timed automata**, a model for representing quantitative constraints on timed systems:
  - useful for modelling resources in timed systems
  - natural (optimization/management) questions have been posed...
  - ... and not all of them have been answered!

Not mentioned here:
- all works on model-checking issues (extensions of CTL, LTL) models based on hybrid automata
- weighted o-minimal hybrid games
- weighted strong reset hybrid games
- various tools have been developed: Uppaal, Uppaal Cora, Uppaal Tiga

Current and further work:
- computation of approximate optimal values
- further investigation of safe games + several cost variables?
- discounted-time optimal games
- link between discounted-time games and mean-cost games?
- computation of equilibria
Conclusion

- **Priced/weighted timed automata**, a model for representing quantitative constraints on timed systems:
  - useful for modelling resources in timed systems
  - natural (optimization/management) questions have been posed...
    ... and not all of them have been answered!

- **Not mentioned here:**
  - all works on model-checking issues (extensions of CTL, LTL)
  - models based on hybrid automata
    - weighted o-minimal hybrid games
    - weighted strong reset hybrid games
  - various tools have been developed:
    Uppaal, Uppaal Cora, Uppaal Tiga
Conclusion

- **Priced/weighted timed automata**, a model for representing quantitative constraints on timed systems:
  - useful for modelling resources in timed systems
  - natural (optimization/management) questions have been posed...
  - ... and not all of them have been answered!

- **Not mentioned here**:
  - all works on model-checking issues (extensions of CTL, LTL)
  - models based on hybrid automata
    - weighted o-minimal hybrid games
    - weighted strong reset hybrid games
  - various tools have been developed:
    - Uppaal, Uppaal Cora, Uppaal Tiga

- **Current and further work**:
  - computation of approximate optimal values
  - further investigation of safe games + several cost variables?
  - discounted-time optimal games
  - link between discounted-time games and mean-cost games?
  - computation of equilibria
  - ...