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?
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

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

Modelling

Controller synthesis

→ modeled as two player games
Timed automata

$x, y : \text{clocks}$

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

$y > 1, \ b, \ x := 0$

---

<table>
<thead>
<tr>
<th>$\ell_0$</th>
<th>$\delta(4.1)$</th>
<th>$\ell_0$</th>
<th>$a$</th>
<th>$\ell_1$</th>
<th>$\delta(1.4)$</th>
<th>$\ell_1$</th>
<th>$b$</th>
<th>$\ell_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>0</td>
<td>4.1</td>
<td>4.1</td>
<td></td>
<td>5.5</td>
<td>1.4</td>
<td>1.4</td>
<td></td>
</tr>
<tr>
<td>$y$</td>
<td>0</td>
<td>4.1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Model of weighted timed automata

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

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

[HSCC’01]
Model of weighted timed automata

- Cost rate
- Discrete cost

\[ P \xrightarrow{\delta(d)} P' \]
\[ g, \ a, \ C := 0 \]

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

\[
\begin{align*}
(\ell, v) & \xrightarrow{\delta(d)} (\ell, v + d) \\
(\ell, v) & \xrightarrow{a} (\ell', v') \quad \text{where} \quad v \models g \\
& \quad v' = [C \leftarrow 0]v \\
& \quad \text{for some } \ell \xrightarrow{g, a, C := 0} \ell'
\end{align*}
\]
Model of weighted timed automata

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

\[
\begin{cases}
\ell, v \xrightarrow{\delta(d)} \ell, v + d \\
\ell, v \xrightarrow{a} (\ell', v') \text{ where } \begin{cases} v \models g \\
v' = [C \leftarrow 0]v \end{cases}
\end{cases}
\]

for some \(\ell \xrightarrow{g,a,C:=0} \ell'\)

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

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

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

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

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?

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

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

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

**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

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

\[ P \rightarrow \text{cost rate} \]

\[ P \rightarrow \text{discrete cost} \]

\[ P' \]

\[ \ell' \]
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.’06]

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

- Model-checking WCTL, an extension of CTL with cost constraints

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

[Brihaye, Bruyère, Raskin – FORMATS+FTRTFT’04]
[Bouyer, Brihaye, Markey – IPL’06]
[Bouyer, Laroussinie, Larsen, Markey, Rasmussen – Subm.’06]
The classical region abstraction

\[ \text{reset to 0} \quad \text{time elapsing} \]
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*:
The corner-point abstraction

**Idea:** reduction to the discrete case
- region abstraction: not sufficient
- corner-point abstraction:

```
  3  0  0  0  0
```

reset to 0

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

Idea: reduction to the discrete case
- region abstraction: not sufficient
- corner-point abstraction:

- time elapsing
  \[ \Rightarrow \]
  reset to 0

\[
\begin{array}{cccc}
0 & 0 & 0 & 0 \\
\\
7 & & & \\
\\
3 & & & \\
\end{array}
\]

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:**

![Diagram of corner-point abstraction]

- time elapsing
  - reset to 0

**This abstraction is correct!**
- for computing optimal paths
- for computing optimal stationary behaviours
Outline

1. Introduction
2. Model-checking weighted timed automata
3. Optimal timed games
4. Conclusion
Decidability of timed games

**Theorem**  
[Henzinger, Kopke 1999]  
Safety and reachability control in timed automata are decidable and EXPTIME-complete.

(\text{the attractor is computable...})

$\rightarrow$ classical regions are sufficient for solving such problems
An example

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

- **$\ell_0$**
  - $x \leq 2$; $c$; $y := 0$
  - cost($\ell_0$) = 5

- **$\ell_1$**
  - $y = 0$

- **$\ell_2$**
  - cost($\ell_2$) = 10
  - $x \geq 2$; $c$; cost = 1

- **$\ell_3$**
  - cost($\ell_3$) = 1
  - $x \geq 2$; $c$; cost = 7
An example

c: controllable action
u: uncontrollable action

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

cost(\ell_0) = 5
y = 0
cost(\ell_2) = 10
x \geq 2; cost = 1
cost(\ell_3) = 1

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

c: controllable action
u: uncontrollable action

(cost(l_0) = 5, x \leq 2; c; y := 0)

(cost(l_1) = 0, y = 0)

(cost(l_2) = 10, x \geq 2; c; cost = 1)

(cost(l_3) = 1, x \geq 2; c; cost = 7)

Question: what is the optimal cost we can ensure in state l_0?

5t + 10(2 − t) + 1
An example

c: controllable action
u: uncontrollable action

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

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

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

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

**Question:** what is the optimal cost we can ensure in state $l_0$?

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

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

cost($l_0$) = 5

$\ell_0 \xrightarrow{x \leq 2; c; y := 0} \ell_1$

$\ell_1 \xrightarrow{y = 0} \ell_2$

cost($\ell_2$) = 10

$\ell_2 \xrightarrow{u} \ell_3$

cost($\ell_3$) = 1

$\ell_3 \xrightarrow{x \geq 2; c; cost = 1} \text{happy face}$

$\ell_3 \xrightarrow{x \geq 2; c; cost = 7} \text{happy face}$

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

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

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

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

\[ \text{cost}(\ell_2) = 10 \]

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

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

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

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

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

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

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

c: controllable action
u: uncontrollable action

\[
\begin{align*}
\ell_0 & \xrightarrow{x \leq 2; \ c} \ell_1 \quad \text{cost}(\ell_0) = 5 \\
\ell_1 & \xrightarrow{y = 0} \ell_2 \\
\ell_2 & \xrightarrow{u} \ell_3 \quad \text{cost}(\ell_2) = 10 \\
\ell_3 & \xrightarrow{x \geq 2; \ c} \ell_4 \quad \text{cost}(\ell_3) = 1
\end{align*}
\]

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

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

→ 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

- \( \ell_0 \rightarrow \ell_1 \) : \( x \leq 2; \ c; \ y := 0 \)

- \( \ell_1 \rightarrow \ell_2 \) : \( y = 0 \)

- \( \ell_2 \rightarrow \ell_3 \) : \( x \geq 2; \ c; \ \text{cost} = 1 \)

- \( \ell_3 \rightarrow \) : \( x \geq 2; \ c; \ \text{cost} = 7 \)

- \( \ell_0 \rightarrow \ell_1 \) : \( \text{cost}(\ell_0) = 5 \)

- \( \ell_1 \rightarrow \ell_2 \) : \( \text{cost}(\ell_1) = 1 \)

- \( \ell_2 \rightarrow \ell_3 \) : \( \text{cost}(\ell_2) = 10 \)

- \( \ell_3 \rightarrow \) : \( \text{cost}(\ell_3) = 1 \)

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

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

**Strategy:** wait in \( \ell_0 \), and when \( t = \frac{4}{3} \), go to \( \ell_1 \)

- How to automatically compute such optimal costs?
- 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, Laroussinie, 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, Laroussinie, Larsen, Markey, Rasmussen – Subm.’06]:
  - with one clock, optimal cost is computable
- [Jurdziński, Trivedi – LICS’06]:
  - optimal mean-cost is computable in a (restrictive) case
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 $l_1$, 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 $\ell_1$, the cost of the run is $2.d + d'$.

$$2.d + d' \leq 2$$
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 $2d + d'$.

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

(accumulated cost) + $d' \leq 2$
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. $x_1 = \frac{1}{2c_1}$
- 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 $\leq 3$, 
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 \( \leq 3 \), 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 $j$

Test($x = 2z, \{y\}$)
Simulation of an incrementation

Instruction $i$: $c_1 + +; \text{goto instruction } j$

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

\[
\begin{align*}
\text{Test}(x = 2z, \{y\})
\end{align*}
\]

\[
\begin{align*}
\left(\begin{array}{c}
c_1 \mapsto x \\
c_2 \mapsto y
\end{array}\right) \\
\left\{\begin{array}{l}
x = \frac{1}{2^{c_1}} \\
y = \frac{1}{3^{c_2}}
\end{array}\right.
\end{align*}
\]

\[
\begin{align*}
\left(\begin{array}{c}
c_1 \mapsto z \\
c_2 \mapsto y
\end{array}\right) \\
\left\{\begin{array}{l}
z = \frac{1}{2^{c_1+1}} \\
y = \frac{1}{3^{c_2}}
\end{array}\right.
\end{align*}
\]
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

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

$y=1, y:=0$  
$x=1, x:=0$  
$z=0$  
$\text{cost}=0$

$y=1, y:=0$  
$z=1, z:=0$  
$\text{cost}=1$

The cost is increased by $x_0$

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

$y=1, y:=0$  
$x=1, x:=0$  
$z:=0$  
$\text{cost}=1$

$y=1, y:=0$  
$z=1, z:=0$  
$\text{cost}=0$

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 $y = 2x$

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
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 \( y = 2x \)

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 \( W_1 \), \( \text{cost} > 3 \)
- if \( y_0 > 2x_0 \), player 2 chooses the second branch: in \( W_2 \), \( \text{cost} > 3 \)
- if \( y_0 = 2x_0 \), in \( W_1 \) or in \( W_2 \), \( \text{cost} = 3 \).
How to get rid of tick clock $u$?

$y=1, y:=0$

$x=1, x:=0$

$z:=0$

$u=1$

$u=1, u:=0$

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:

- no cost is accumulated in $D$-states
- 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 $\frac{1}{2\pi}$
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 $\frac{1}{2^m}$
  - the value of $y$ in $D$ is of the form $\frac{1}{3^m}$
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 $\frac{1}{2n}$
  - the value of $y$ in $D$ is of the form $\frac{1}{3m}$

Optimal timed games
Checking that $x$ is of the form $\frac{1}{2^n}$
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
- ...