Priced Timed Automata
Optimal Scheduling

Kim G. Larsen
CISS – Aalborg University
DENMARK
Overview

- Timed Automata
  - Scheduling

- Priced Timed Automata
  - Optimal Reachability
  - Optimal Infinite Scheduling
  - Multi Objectives

- Energy Automata
Real Time Scheduling

- Only 1 “Pass”
- Cheat is possible
  (drive close to car with “Pass”)

SAFE CAN THEY MAKE IT TO SAFE WITHIN 70 MINUTES ???

UNSAFE

Crossing Times

5
10 Pass
20
25
Let us play!

Solving scheduling problems using Uppaal

A number of cars are to pass a bridge. There is a toll for passing the bridge – and a device (known as the "Brofizz" or "EasyPass") must be used in order to pass the bridge.

There is only one Brofizz available to the cars – but luckily the toll booth system can be cheated if two cars drive close to each other. Only cars from the side at which Brofizz is located can pass the bridge. The toll booth at the side at which the Brofizz is located is coloured green.

All cars must pass the bridge within a given time limit (shown at the center of the screen). Each car spends a given number of minutes passing the bridge. This scheduling/bridge problem can be solved using the Uppaal tool.

Using the buttons above you can:

- Configure:
  - Set the number of cars, their speed, and the time limit
- Interact:
  - Try to solve the problem manually
- Find some solution:
  - Use Uppaal to solve the problem and display the solution
- Find best solution:
  - Use Uppaal to find the best solution to the problem
Real Time Scheduling

Solve Scheduling Problem using UPPAAL

ARTIST Design PhD School, Beijing, 2011

Kim Larsen [5]
Resources & Tasks

Resource

Idle
use?
x:=0

InUse

x>=B

x<=B

Synchronization

done!

Task

Init

use!

Using

B:=6
done?

Shared variable

Done

ARTIST Design PhD School, Beijing, 2011

Kim Larsen [6]
Task Graph Scheduling – Example

Compute:
\[(D \times (C \times (A + B)) + ((A + B) + (C \times D))\]
using 2 processors

P1 (fast)
P2 (slow)

13 pico-sec !!
Compute:

\[(D \times (C \times (A + B)) + ((A + B) + (C \times D))\]

using 2 processors

P1 (fast)

P2 (slow)

12 pico-sec

OPTIMAL!!
Task Graph Scheduling – Example

Compute:

\((D \times (C \times (A + B))) + ((A + B) + (C \times D))\)

A

B

C

D

\[ \begin{align*}
A & \rightarrow 1 \\
B & \rightarrow 2 \\
C & \rightarrow 3 \\
D & \rightarrow 4
\end{align*} \]

\[ \begin{align*}
\text{P1 (fast)} \\
\text{P2 (slow)}
\end{align*} \]
Task Graph Scheduling – Example

Compute:

\((D \times (C \times (A + B)) + ((A + B) + (C \times D)))\)

using 2 processors

\(A\)
\(B\)
\(C\)
\(D\)

\(P1\) (fast)
\(P2\) (slow)

\(C\)

\(E<>\) (Task1.End and … and Task6.End)
Experimental Results

<table>
<thead>
<tr>
<th>name</th>
<th>#tasks</th>
<th>#chains</th>
<th># machines</th>
<th>optimal</th>
<th>TA</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>437</td>
<td>125</td>
<td>4</td>
<td>1178</td>
<td>1182</td>
</tr>
<tr>
<td>000</td>
<td>452</td>
<td>43</td>
<td>20</td>
<td>537</td>
<td>537</td>
</tr>
<tr>
<td>018</td>
<td>730</td>
<td>175</td>
<td>10</td>
<td>700</td>
<td>704</td>
</tr>
<tr>
<td>074</td>
<td>1007</td>
<td>66</td>
<td>12</td>
<td>891</td>
<td>894</td>
</tr>
<tr>
<td>021</td>
<td>1145</td>
<td>88</td>
<td>20</td>
<td>605</td>
<td>612</td>
</tr>
<tr>
<td>228</td>
<td>1187</td>
<td>293</td>
<td>8</td>
<td>1570</td>
<td>1574</td>
</tr>
<tr>
<td>071</td>
<td>1193</td>
<td>124</td>
<td>20</td>
<td>629</td>
<td>634</td>
</tr>
<tr>
<td>271</td>
<td>1348</td>
<td>127</td>
<td>12</td>
<td>1163</td>
<td>1164</td>
</tr>
<tr>
<td>237</td>
<td>1566</td>
<td>152</td>
<td>12</td>
<td>1340</td>
<td>1342</td>
</tr>
<tr>
<td>231</td>
<td>1664</td>
<td>101</td>
<td>16</td>
<td>t.o.</td>
<td>1137</td>
</tr>
<tr>
<td>235</td>
<td>1782</td>
<td>218</td>
<td>16</td>
<td>t.o.</td>
<td>1150</td>
</tr>
<tr>
<td>233</td>
<td>1980</td>
<td>207</td>
<td>19</td>
<td>1118</td>
<td>1121</td>
</tr>
<tr>
<td>294</td>
<td>2014</td>
<td>141</td>
<td>17</td>
<td>1257</td>
<td>1261</td>
</tr>
<tr>
<td>295</td>
<td>2168</td>
<td>965</td>
<td>18</td>
<td>1318</td>
<td>1322</td>
</tr>
<tr>
<td>292</td>
<td>2333</td>
<td>318</td>
<td>3</td>
<td>8009</td>
<td>8009</td>
</tr>
<tr>
<td>298</td>
<td>2399</td>
<td>303</td>
<td>10</td>
<td>2471</td>
<td>2473</td>
</tr>
</tbody>
</table>

Symbolic A*
Branch-&-Bound
60 sec

Abdeddaïm, Kerbaa, Maler
EXAMPLE: Optimal rescue plan for cars with different subscription rates for city driving!

**Optimal Plan** has accumulated **Cost** = 195 and **Total Time** = 65!
## Experiments

<table>
<thead>
<tr>
<th>COST-rates</th>
<th>SCHEDULE</th>
<th>COST</th>
<th>TIME</th>
<th>#Expl</th>
<th>#Pop’d</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>C</td>
<td>B</td>
<td>D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Min Time</td>
<td>CG&gt; G&lt; BD&gt; C&lt; CG&gt;</td>
<td>60</td>
<td></td>
<td>1762</td>
<td>2638</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CG&gt; G&lt; BG&gt; G&lt; GD&gt;</td>
<td>55</td>
<td>65</td>
<td>252</td>
<td>378</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>3</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>GD&gt; G&lt; CG&gt; G&lt; BG&gt;</td>
<td>195</td>
<td>65</td>
<td>149</td>
<td>233</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CG&gt; G&lt; BD&gt; C&lt; CG&gt;</td>
<td>140</td>
<td>60</td>
<td>232</td>
<td>350</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CD&gt; C&lt; CB&gt; C&lt; CG&gt;</td>
<td>170</td>
<td>65</td>
<td>263</td>
<td>408</td>
</tr>
<tr>
<td>1</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>BD&gt; B&lt; CB&gt; C&lt; CG&gt;</td>
<td>975</td>
<td>85</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ARTIST Design PhD School, Beijing, 2011

Kim Larsen [14]
Compute:
\[(D \times (C \times (A + B)) + ((A + B) + (C \times D))\]

using 2 processors

**P1** (fast)

**P2** (slow)

**ENERGY:**

<table>
<thead>
<tr>
<th></th>
<th>Idle</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>P1</strong></td>
<td>10W</td>
<td>90W</td>
</tr>
<tr>
<td><strong>P2</strong></td>
<td>20W</td>
<td>30W</td>
</tr>
</tbody>
</table>

Energy: **1.39 nano-joule !!**
Task Graph Scheduling – Revisited

Compute:
\[(D \times (C \times (A + B)) + ((A + B) + (C \times D))\]

using 2 processors

P1 (fast)
- +: 2ps
- *: 3ps
- Idle: 10W
- In use: 90W

P2 (slow)
- +: 5ps
- *: 7ps
- Idle: 20W
- In use: 30W

Energy: 1.32 nano-joule

OPTIMAL!!
Task Graph Scheduling – Revisited

Compute :
\((D \times (C \times (A + B)) + ((A + B) + (C \times D)))\)

\[\begin{align*}
\text{A} &\quad 1 \\
\text{B} &\quad 2 \\
\text{C} &\quad \star \\
\text{D} &\quad \\
\end{align*}\]

\[\begin{align*}
\text{Compute} &\quad 21 \\
\text{processors} &\quad 2 \\
\text{A} &\quad 3 \\
\text{B} &\quad 4 \\
\text{C} &\quad 10W \\
\text{D} &\quad 90W \\
\text{P1 (fast)} &\quad 5ps \\
\text{P2 (slow)} &\quad 7ps \\
\text{ENERGY:} &\quad 1 \text{W}, 3 \text{W}, 4 \text{W} \\
\text{ARTIST Design PhD School, Beijing, 2011} &\quad \text{Kim Larsen [17]} \\
\end{align*}\]
A simple example

Observer variable $C$:

\( \frac{dC}{dt} = +10 \)

\( C := C + 1 \)

\( C := C + 7 \)

\( x=2 \)

\( y=0 \)

\( x<=2 \)

\( y:=0 \)

\( \frac{dC}{dt} = +5 \)

\( \frac{dC}{dt} = +1 \)

\((\ell_0, [0, 0]) \xrightarrow{1.9}{9.5} (\ell_0, [1.9, 1.9]) \rightarrow_0 (\ell_1, [1.9, 0]) \rightarrow_0 (\ell_2, [1.9, 0]) \xrightarrow{0.1}{1.0} (\ell_2, [2, 0.1]) \rightarrow_\gamma (\ell_4, [2, 0.1]) \)

\( \sum C_i = 16.6 \)

\((\ell_0, [0, 0]) \xrightarrow{1.2}{6.0} (\ell_0, [1.2, 1.2]) \rightarrow_0 (\ell_1, [1.2, 0]) \rightarrow_0 (\ell_3, [1.2, 0]) \xrightarrow{0.8}{8.0} (\ell_3, [2, 0.8]) \rightarrow_1 (\ell_4, [2, 0.8]) \)

\( \sum C_i = 15.0 \)
A simple example

Q: What is cheapest cost for reaching $l_4$?

\[ \inf_{0 \leq t \leq 2} \min \{5t + 10(2 - t) + 1, 5t + (2 - t) + 4\} = 9 \]

=> strategy: leave immediately $l_0$, go to $l_3$, and wait there 2 t.u.
Corner Point Regions

**THM [Behrmann, Fehnker ..01] [Alur,Torre,Pappas 01]**
Optimal reachability is decidable for PTA

**THM [Bouyer, Brojaue, Briuere, Raskin 07]**
Optimal reachability is PSPACE-complete for PTA
Priced Zones

A zone $Z$:
$$1 \leq x \leq 2 \quad \land \quad 0 \leq y \leq 2 \quad \land \quad x - y \geq 0$$

A cost function $C$:
$$C(x,y) = 2 \cdot x - 1 \cdot y + 3$$
Priced Zones – Reset

A zone $Z$:

$1 \leq x \leq 2 \land 0 \leq y \leq 2 \land x - y \geq 0$

A cost function $C$:

$C(x,y) = 2 \cdot x - 1 \cdot y + 3$

$Z[x=0]$:

$x=0 \land 0 \leq y \leq 2$

$C = 1 \cdot y + 3$

$C = -1 \cdot y + 5$
Symbolic Branch & Bound Algorithm

\[
\text{Cost} := \infty \\
\text{Passed} := \emptyset \\
\text{Waiting} := \{(l_0, Z_0)\} \\
\textbf{while} \ \text{Waiting} \neq \emptyset \ \textbf{do} \\
\quad \textbf{select} \ (l, Z) \ \textbf{from} \ \text{Waiting} \\
\quad \textbf{if} \ l = l_g \ \text{and} \ \min\text{Cost}(Z) < \text{Cost} \ \textbf{then} \\
\quad \quad \text{Cost} := \min\text{Cost}(Z) \\
\quad \textbf{if} \ \min\text{Cost}(Z) + \text{Rem}_{(l,Z)} \geq \text{Cost} \ \textbf{then} \ \text{break} \\
\quad \textbf{if} \ \text{for all} \ (l', Z') \ \text{in} \ \text{Passed}: \ Z' \not\leq Z \ \textbf{then} \\
\quad \quad \text{add} \ (l, Z) \ \text{to} \ \text{Passed} \\
\quad \quad \text{add} \ \text{all} \ (l', Z') \ \text{with} \ (l, Z) \rightarrow (l', Z') \\
\textbf{return} \ \text{Cost} \\
\]

\[Z' \leq Z\] is bigger & cheaper than \(Z\)

\[\leq\] is a well-quasi ordering which guarantees termination!
Example: Aircraft Landing

Planes have to keep separation distance to avoid turbulences caused by preceding planes.

- **E**: earliest landing time
- **T**: target time
- **L**: latest time
- **e**: cost rate for being early
- **l**: cost rate for being late
- **d**: fixed cost for being late

\[
e^*(T-t) \quad \text{cost} \quad \text{d} + l^*(t-T)
\]
Example: Aircraft Landing

Planes have to keep separation distance to avoid turbulences caused by preceding planes.

4 earliest landing time
5 target time
9 latest time
3 cost rate for being early
1 cost rate for being late
2 fixed cost for being late
### Aircraft Landing

**Source of examples:**
Baesley et al’2000

<table>
<thead>
<tr>
<th>problem instance</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>number of planes</td>
<td>10</td>
<td>15</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>30</td>
<td>44</td>
</tr>
<tr>
<td>number of types</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>optimal value</td>
<td>700</td>
<td>1480</td>
<td>820</td>
<td>2520</td>
<td>3100</td>
<td>24442</td>
<td>1550</td>
</tr>
<tr>
<td>explored states</td>
<td>481</td>
<td>2149</td>
<td>920</td>
<td>5693</td>
<td>15069</td>
<td>122</td>
<td>662</td>
</tr>
<tr>
<td>cpu time (secs)</td>
<td>4.19</td>
<td>25.30</td>
<td>11.05</td>
<td>87.67</td>
<td>220.22</td>
<td>0.60</td>
<td>4.27</td>
</tr>
<tr>
<td>optimal value</td>
<td>90</td>
<td>210</td>
<td>60</td>
<td>640</td>
<td>650</td>
<td>554</td>
<td>0</td>
</tr>
<tr>
<td>explored states</td>
<td>1218</td>
<td>1797</td>
<td>669</td>
<td>28821</td>
<td>47993</td>
<td>9035</td>
<td>92</td>
</tr>
<tr>
<td>cpu time (secs)</td>
<td>17.87</td>
<td>39.92</td>
<td>11.02</td>
<td>755.84</td>
<td>1085.08</td>
<td>123.72</td>
<td>1.06</td>
</tr>
<tr>
<td>optimal value</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>130</td>
<td>170</td>
<td>0</td>
<td>N/A</td>
</tr>
<tr>
<td>explored states</td>
<td>24</td>
<td>46</td>
<td>84</td>
<td>207715</td>
<td>189602</td>
<td>62</td>
<td>N/A</td>
</tr>
<tr>
<td>cpu time (secs)</td>
<td>0.36</td>
<td>0.70</td>
<td>1.71</td>
<td>14786.19</td>
<td>12461.47</td>
<td>0.68</td>
<td>N/A</td>
</tr>
<tr>
<td>optional value</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>0</td>
<td>0</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>explored states</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>65</td>
<td>64</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>cpu time (secs)</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>1.97</td>
<td>1.53</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Symbolic Branch & Bound Algorithm

Cost := ∞
Passed := ∅
Waiting := {(l₀, Z₀)}
while Waiting ≠ ∅ do
  select (l, Z) from Waiting
  if l = l_g and \( \text{minCost}(Z) < \text{Cost} \) then
    Cost := minCost(Z)
  if \( \text{minCost}(Z) + \text{Rem}(l, Z) \geq \text{Cost} \) then break
  if for all \((l', Z')\) in Passed: \( Z' \notin Z \) then
    add \((l, Z)\) to Passed
  add all \((l', Z')\) with \((l, Z) \rightarrow (l', Z')\) to Waiting
return Cost

Zone based
Linear Programming Problems
\( \rightarrow \) (dualize)
Min Cost Flow

Kim Larsen [27]
## Aircraft Landing (revisited)

### Using MCF/Netsimplex

<table>
<thead>
<tr>
<th>RW</th>
<th>Planes</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>20</th>
<th>20</th>
<th>30</th>
<th>44</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Types</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>simplex</td>
<td>0.844s</td>
<td>5.210s</td>
<td>2.135s</td>
<td>17.888s</td>
<td>44.878s</td>
<td>0.451s</td>
<td>0.670s</td>
</tr>
<tr>
<td></td>
<td>netsimplex</td>
<td>0.156s</td>
<td>0.657s</td>
<td>0.369s</td>
<td>2.363s</td>
<td>5.503s</td>
<td>0.127s</td>
<td>0.322s</td>
</tr>
<tr>
<td></td>
<td>factor</td>
<td>5.41</td>
<td>7.93</td>
<td>5.79</td>
<td>7.57</td>
<td>8.16</td>
<td>3.55</td>
<td>2.08</td>
</tr>
<tr>
<td>2</td>
<td>simplex</td>
<td>2.577s</td>
<td>7.436s</td>
<td>2.175s</td>
<td>94.357s</td>
<td>120.004s</td>
<td>2.322s</td>
<td>0.264s</td>
</tr>
<tr>
<td></td>
<td>netsimplex</td>
<td>0.332s</td>
<td>1.036s</td>
<td>0.436s</td>
<td>13.376s</td>
<td>18.033s</td>
<td>0.600s</td>
<td>0.179s</td>
</tr>
<tr>
<td></td>
<td>factor</td>
<td>8.00</td>
<td>7.18</td>
<td>4.99</td>
<td>7.054</td>
<td>6.65</td>
<td>3.87</td>
<td>1.474</td>
</tr>
<tr>
<td>3</td>
<td>simplex</td>
<td>0.120s</td>
<td>0.181s</td>
<td>0.357s</td>
<td>740.100s</td>
<td>516.678s</td>
<td>0.166s</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>netsimplex</td>
<td>0.064s</td>
<td>0.104s</td>
<td>0.129s</td>
<td>170.176s</td>
<td>124.805s</td>
<td>0.079s</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>factor</td>
<td>1.87</td>
<td>1.74</td>
<td>2.77</td>
<td>4.34</td>
<td>4.14</td>
<td>2.10</td>
<td>N/A</td>
</tr>
<tr>
<td>4</td>
<td>simplex</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>1.603s</td>
<td>0.318s</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>netsimplex</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>0.378s</td>
<td>0.093s</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>factor</td>
<td>4.24</td>
<td>3.42</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Optimal Schedule

\[(\ell_0, [0, 0]) \xRightarrow{1.2}{6.0} (\ell_0, [1.2, 1.2]) \rightarrow_0 (\ell_1, [1.2, 0]) \rightarrow_0 \]
\[ (\ell_3, [1.2, 0]) \xRightarrow{0.8}{8.0} (\ell_3, [2, 0.8]) \rightarrow_1 (\ell_4, [2, 0.8]) \rightarrow_2 (\ell_0, [0, 0]) \]

\[\sum_i C_i / \sum_i t_i = 17/2 = 8.5\]
EXAMPLE: Optimal WORK plan for cars with different subscription rates for city driving!

maximal 100 min. at each location

process Torch
0

free1
L:= -L+1
release?

free2

one
release

take?

Golf

y<=100 unsafe
L := 0
0
y := 0

ty >= 5
release!
9
take!
y := 100

ty >= 5
release!
y := 0

Citroen

y<=100 unsafe
L := 0
2
y := 0

ty >= 10
release!

BMW

y<=100 unsafe
L := 0
3
y := 0

ty >= 20
release!
take!
y := 100

Datsun

y<=100 unsafe
L := 0
10
y := 0

ty >= 25
release!
take!
y := 100

UCb
Workplan I

Value of workplan:

\[
\frac{(4 \times 300)}{90} = 13.33
\]
Workplan II

Value of workplan:

560 / 100 = 5.6
Optimal Infinite Scheduling

Maximize throughput:
i.e. maximize Reward / Time in the long run!
Optimal **Infinite** Scheduling

Minimize Energy Consumption:

i.e. minimize Cost / Time in the long run
Optimal **Infinite** Scheduling

Maximize throughput:

i.e. maximize **Reward / Cost** in the long run
Mean Pay-Off Optimality

Bouyer, Brinksma, Larsen: HSCC04, FMSD07

**THM:** The mean-pay off optimization problem is decidable (and PSPACE-complete) for PTA. Corner Point Abstract Sound & Complete

Optimal Schedule $\sigma^*$: $\text{val}(\sigma^*) = \inf_\sigma \text{val}(\sigma)$
Discount Optimality \( \lambda < 1 \): discounting factor

Larsen, Fahrenberg: INFINITY’08

\[ \text{Cost of time } t_n \]

\[ \text{Time of step } n \]

\[ c(t_1), c(t_2), c(t_3), \ldots, c(t_n) \]

\[ t_1, t_2, t_3, \ldots, t_n \]

\[ \sigma \]

**THM:** The discount optimization problem is decidable for PTA. Corner Point Abstract Sound & Complete

Value of path \( \sigma \): \( \text{val}(\sigma) = \int_{t=0}^{t=\infty} c(t) \lambda^t dt \)

Optimal Schedule \( \sigma^* \): \( \text{val}(\sigma^*) = \inf_{\sigma} \text{val}(\sigma) \)

ARTIST Design PhD School, Beijing, 2011

Kim Larsen [37]
Soundness of Corner Point Abstraction

Lemma
Let $Z$ be a (bounded, closed) zone and let $f$ be a (well-defined) function over $Z$ defined by:

$$f : (t_1, \ldots, t_n) \mapsto \frac{a_1 t_1 + \cdots + a_n t_n + a}{c_1 t_1 + \cdots + c_n t_n + d}$$

then $\inf_Z f$ is obtained at a corner-point of $Z$ (with integer coefficients).

Lemma
Let $Z$ be a (bounded, closed) zone and let $f$ be a function over $Z$ defined by:

$$f : (t_1, \ldots, t_n) \mapsto a_1 \lambda^{t_1} + \cdots + a_n \lambda^{t_n} + a$$

then $\inf_Z f$ is obtained at a corner-point of $Z$ (with integer coefficients).
Application
Dynamic Voltage Scaling
Multiple Objective Scheduling

The Pareto Frontier for Reachability in Multi Priced Timed Automata is computable

[Larsen&Rasmussen: FoSSaCS05]
"Experimental" Results

Warehouse
iTunes

ARTIST Design PhD School, Beijing, 2011
Kim Larsen [41]
"Experimental" Results
Energy Automata
In some cases, resources can both be consumed and regained.

The aim is then to keep the level of resources within given bounds.
Consuming & Harvesting Energy

Maximize throughput while respecting: $0 \leq E \leq \text{MAX}$

ARTIST Design PhD School, Beijing, 2011

Kim Larsen [45]
Energy Constrains

- Energy is not only consumed but may also be regained
- The aim is to **continuously** satisfy some energy constraints

![Diagram of a model with states and transitions]

**lower-weak-upper-bound problem**
## Results (so far)

### Untimed

<table>
<thead>
<tr>
<th>games</th>
<th>existential problem</th>
<th>universal problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L$</td>
<td>$\mathbb{P} \cap \text{coUP}$</td>
<td>$\mathbb{P}$</td>
</tr>
<tr>
<td>$L+W$</td>
<td>$\mathbb{NP} \cap \text{coNP}$</td>
<td>$\mathbb{P}$</td>
</tr>
<tr>
<td>$L+U$</td>
<td>$\text{EXPTIME-}$</td>
<td>$\mathbb{PSPACE}$</td>
</tr>
</tbody>
</table>

### 1 Clock

<table>
<thead>
<tr>
<th>games</th>
<th>existential problem</th>
<th>universal problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L$</td>
<td>?</td>
<td>$\mathbb{P}$</td>
</tr>
<tr>
<td>$L+W$</td>
<td>?</td>
<td>$\mathbb{P}$</td>
</tr>
<tr>
<td>$L+U$</td>
<td>undecidable</td>
<td>?</td>
</tr>
</tbody>
</table>

*Corner Point Abstraction Suffice*
Discrete Updates on Edges

Corner Point Abstraction suffice

Corner Point Abstraction NOT Adequate
New Approach: Energy Functions

- Maximize energy along paths
- Use this information to solve general problem

\[ w_{in} \quad 0 \xrightarrow{c=0} 2 \xrightarrow{-2} 5 \xrightarrow{-1} 7 \xrightarrow{-5} 9 \xrightarrow{c=1} 0 \quad w_{out} \]

<table>
<thead>
<tr>
<th>point</th>
<th>( w_{in} )</th>
<th>( w_{out} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \alpha )</td>
<td>64/35</td>
<td>0</td>
</tr>
<tr>
<td>( \beta )</td>
<td>2</td>
<td>27/35</td>
</tr>
<tr>
<td>( \gamma )</td>
<td>3</td>
<td>18/7</td>
</tr>
<tr>
<td>( \delta )</td>
<td>8</td>
<td>9</td>
</tr>
</tbody>
</table>
Energy Function

\[
\frac{dE}{dt} = 2 \quad \frac{dE}{dt} = 4
\]

**General Strategy**
Spend just enough time to survive the next negative update
Exponential PTA

\[ \frac{dE}{dt} = 2E \quad \frac{dE}{dt} = 4E \]

**General Strategy**
Spend just enough time to survive the next negative update so that after next negative update there is a certain positive amount!

Minimal Fixpoint:

\[ \frac{3}{e^2 - 1} \approx 0.47 \]
**Exponential PTA**

\[ \frac{dE}{dt} = 2E \]

Thm [HSCC10]:
Lower-bound problem is decidable for linear and exponential 1-clock PTAs with negative discrete updates.

- \( f : x \mapsto \alpha \cdot x^r + \beta \) where \( r \) is rational
- \( \frac{df}{dt} \geq 1 \)

Closed under max and composition. Least fixed point computable.
Conclusion

- Priced Timed Automata a uniform framework for modeling and solving dynamic resource allocation problems!
- Not mentioned here:
  - Model Checking Issues (ext. of CTL and LTL).
- Future work:
  - Zone-based algorithm for optimal infinite runs.
  - Approximate solutions for priced timed games to circumvent undecidability issues.
  - Open problems for Energy Automata.
  - Approximate algorithms for optimal reachability