# Priced Timed Automata and Timed Games

Kim G. Larsen
Aalborg University, DENMARK

# Scheduling and Synthesis

Priced Timed Automata

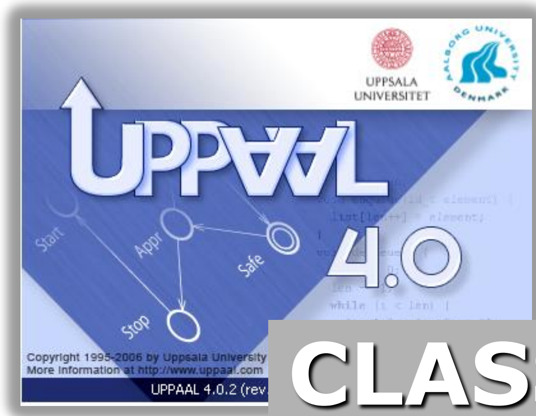
Timed Games

Kim G. Larsen
Aalborg University, DENMARK

# Overview

- Timed Automata & UPPAAL
- Symbolic Verification &
  UPPAAL Engine, Options
- Priced Timed Automata
  and Timed Games
- Stochastic Timed Automata
  Statistical Model Checking

(Lecture+Exercise)$^4$




CLASSIC

CORA

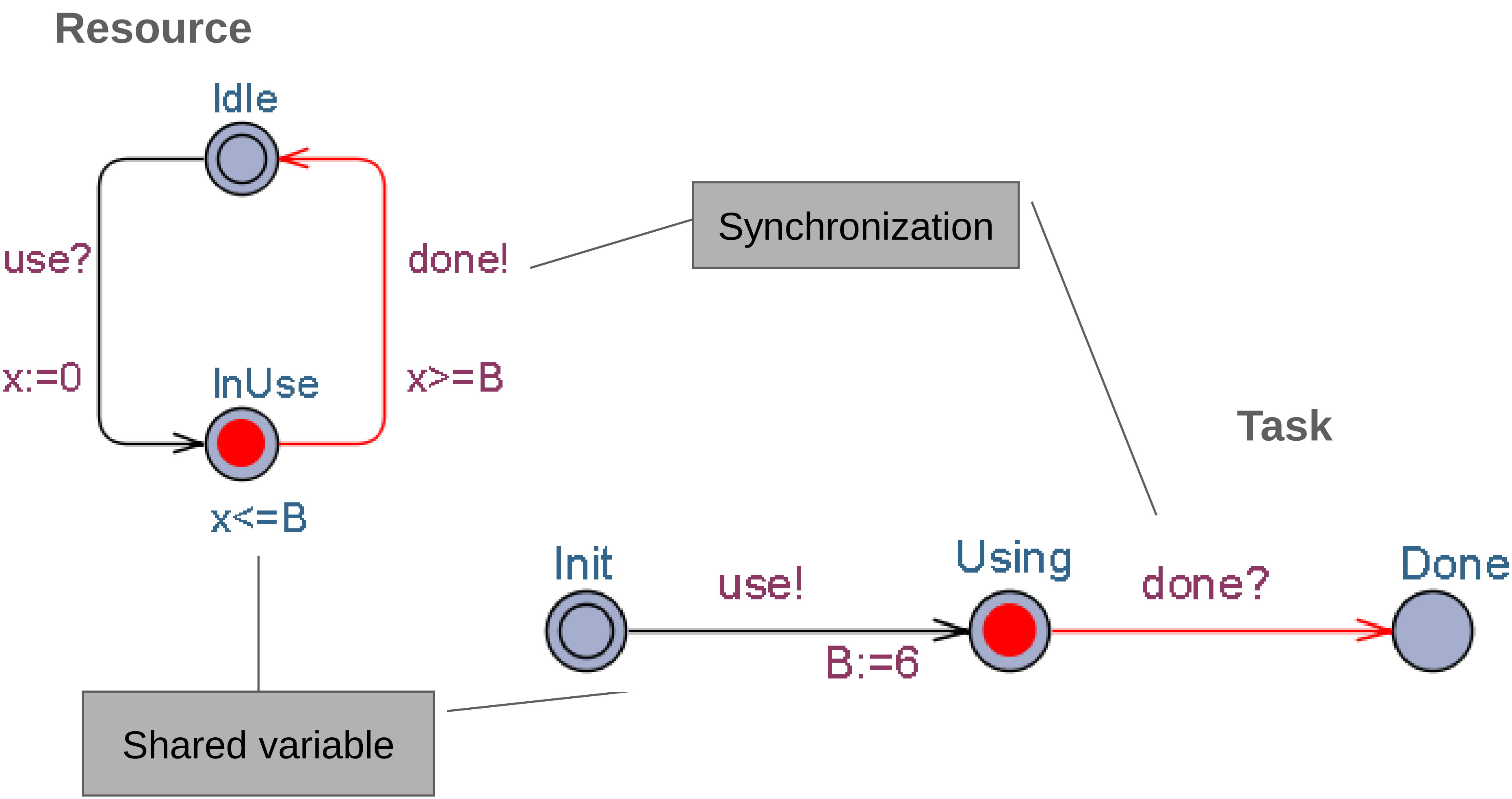
TIGA

ECDAR

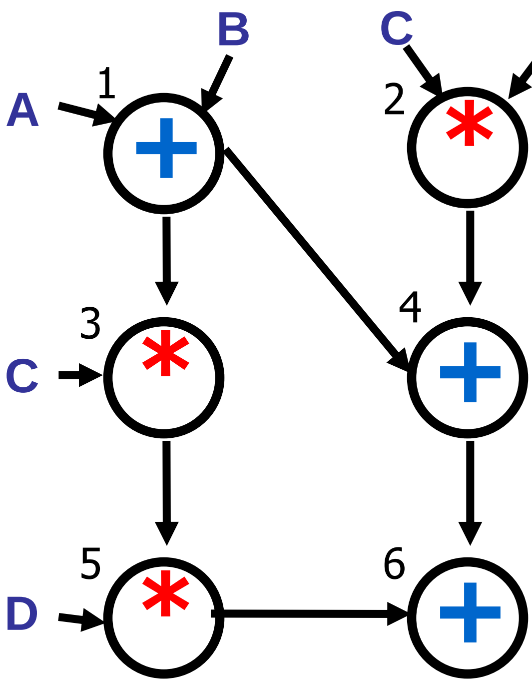
SMC

TRON

# Resources & Tasks



# Task Graph Scheduling – Example




**Compute :**

(D * ( C * ( A + B )) + (( A + B ) + ( C * D ))

4

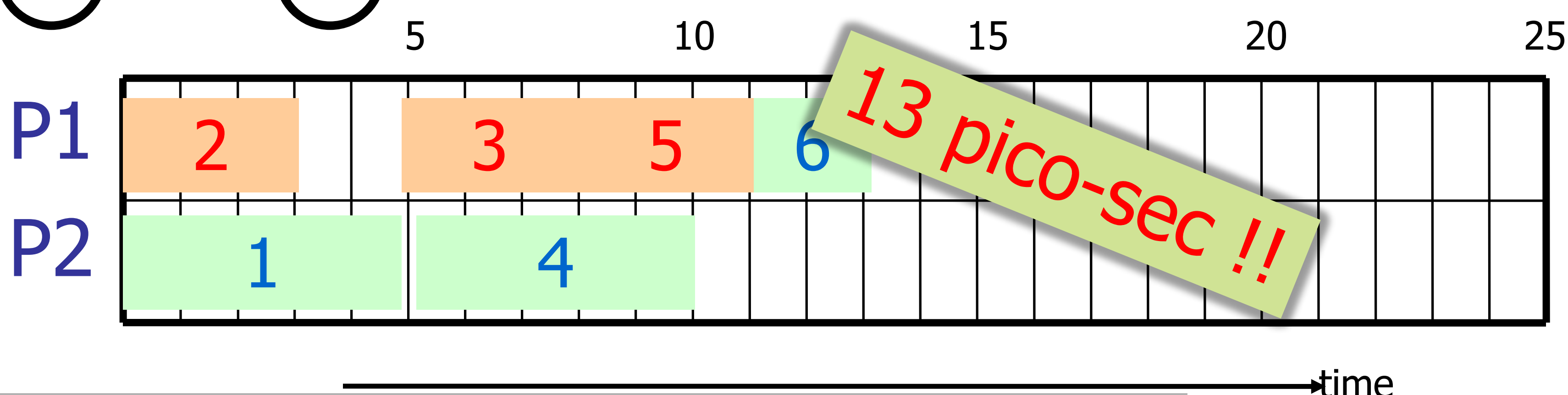
**using 2 processors**

P1 (fast)

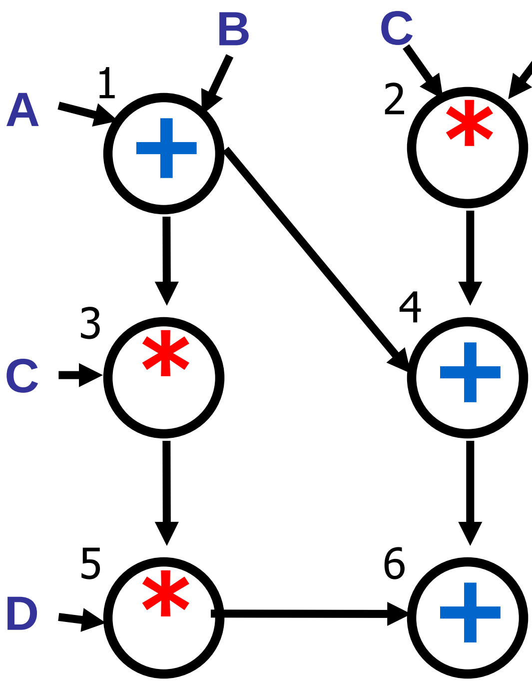

| | |
|---|---|
| + | 2ps |
| * | 3ps |

P2 (slow)

| | |
|---|---|
| + | 5ps |
| * | 7ps |



# Task Graph Scheduling – Example




**Compute :**

**(D * ( C * ( A + B )) + (( A + B ) + ( C * D ))**

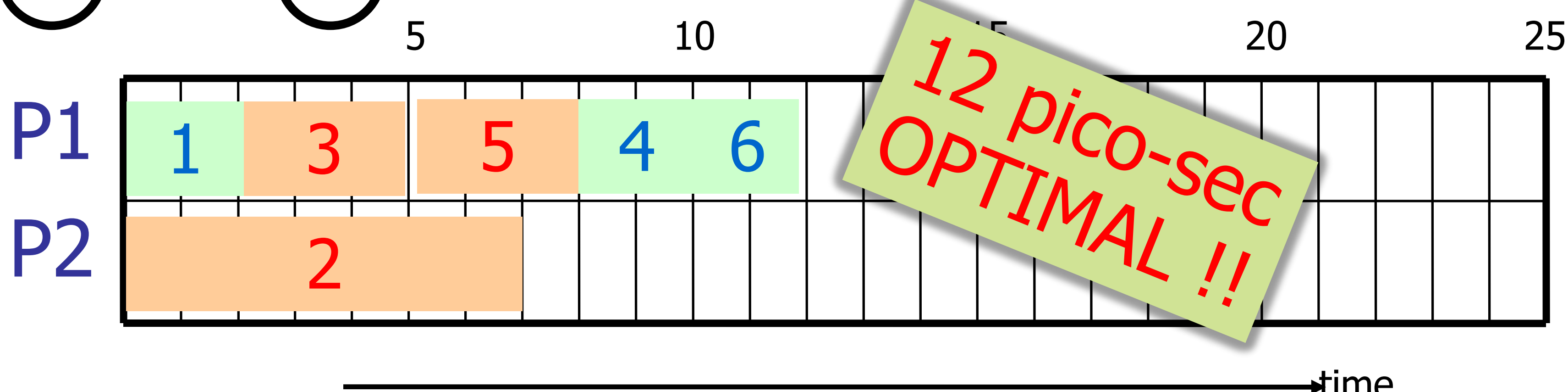
**using 2 processors**

P1 (fast)


| | |
|---|---|
| + | 2ps |
| * | 3ps |

P2 (slow)

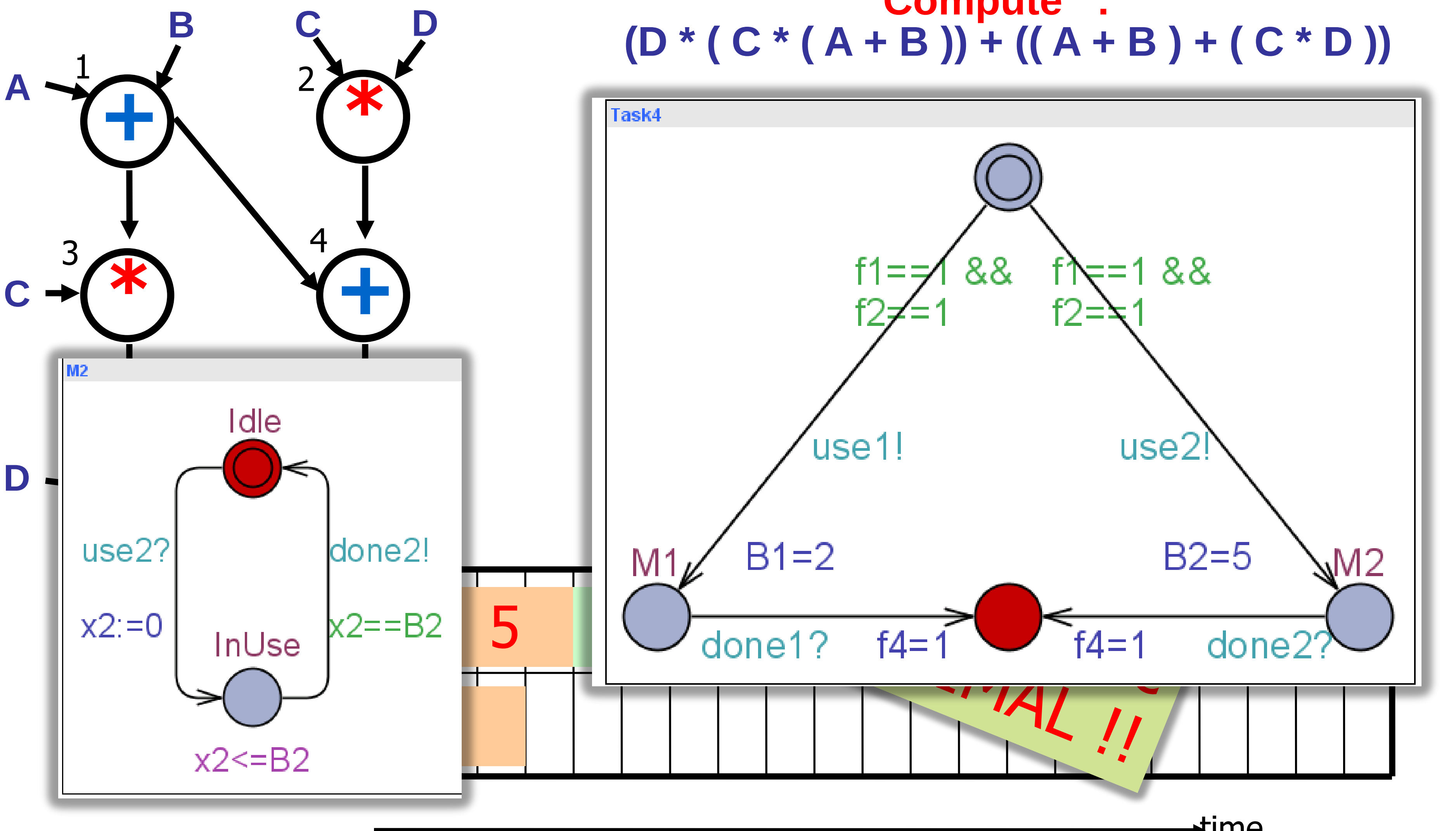
| | |
|---|---|
| + | 5ps |
| * | 7ps |



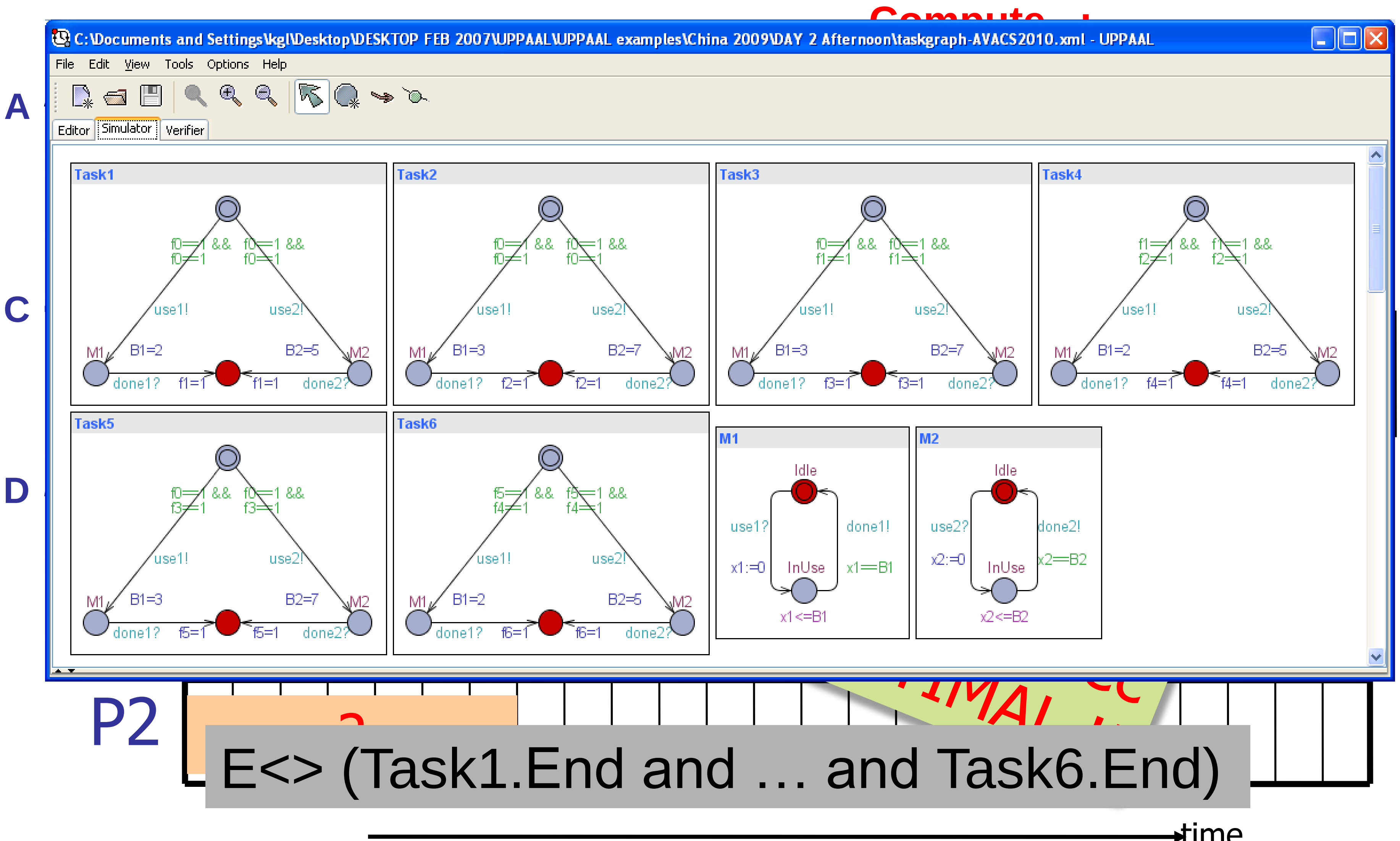
# Task Graph Scheduling – Example

**Compute :**

**(D * ( C * ( A + B )) + (( A + B ) + ( C * D ))**



# Task Graph Scheduling – Example




E<> (Task1.End and … and Task6.End)

# Experimental Results

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

AMETIST
advanced methods for timed systems

Symbolic A*
Branch-&-Bound
60 sec

Abdeddaïm, Kerbaa, Maler

# Jobshop Scheduling

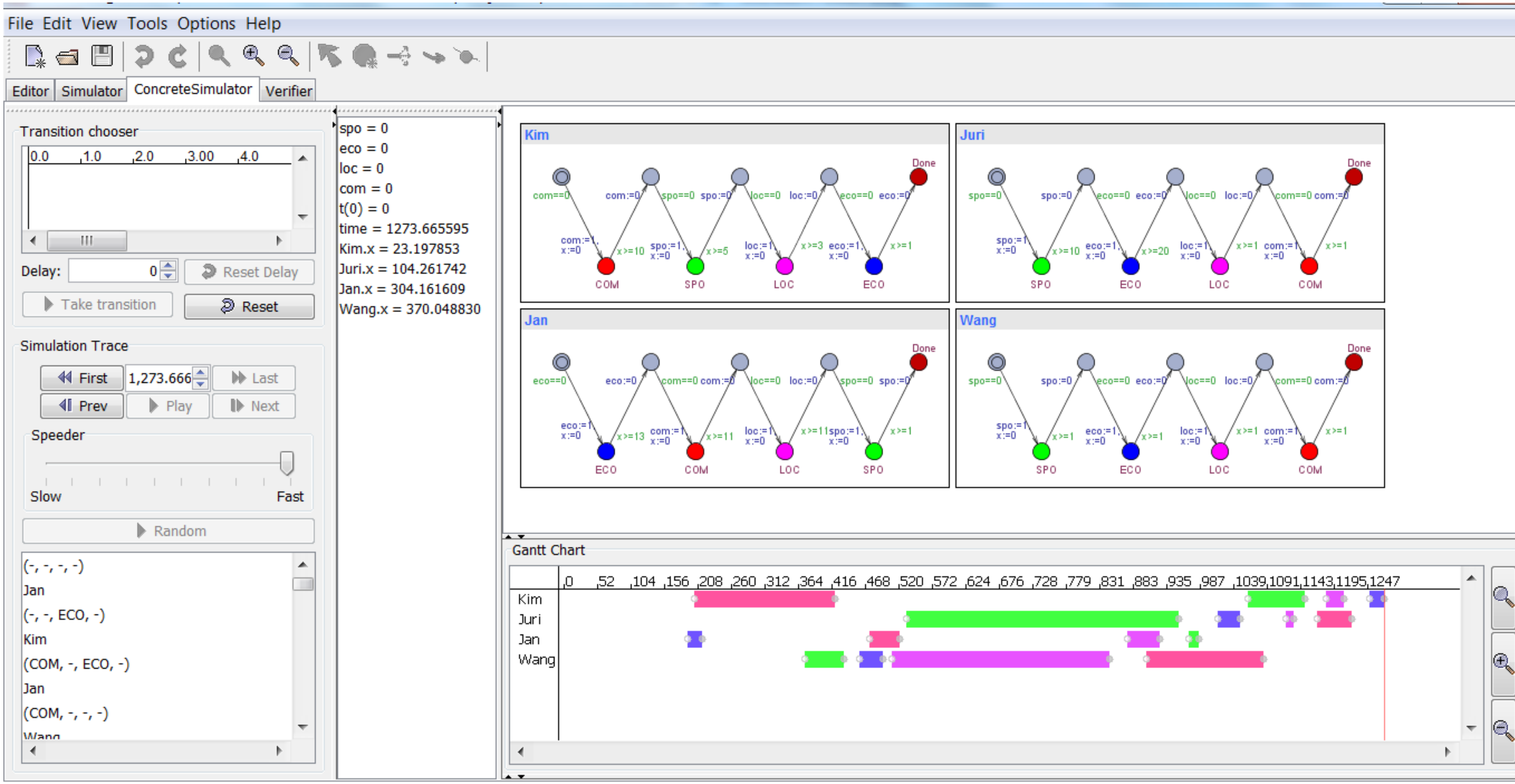
[TACAS'2001]

| | Sport | Economy | Local News | Comic Stip |
|---|---|---|---|---|
| **Kim** | 2. 5 min | 4. 1 min | 3. 3 min | 1. 10 min |
| **Jüri** | 1. 10 min | 2. 20 min | 3. 1 min | 4. 1 min |
| **Jan** | 4. 1 min | 1. 13 min | 3. 11 min | 2. 11 min |
| **Wang** | 1. 1 min | 2. 1 min | 3. 1 min | 4. 1 min |

**Problem:** compute the minimal **MAKESPAN**

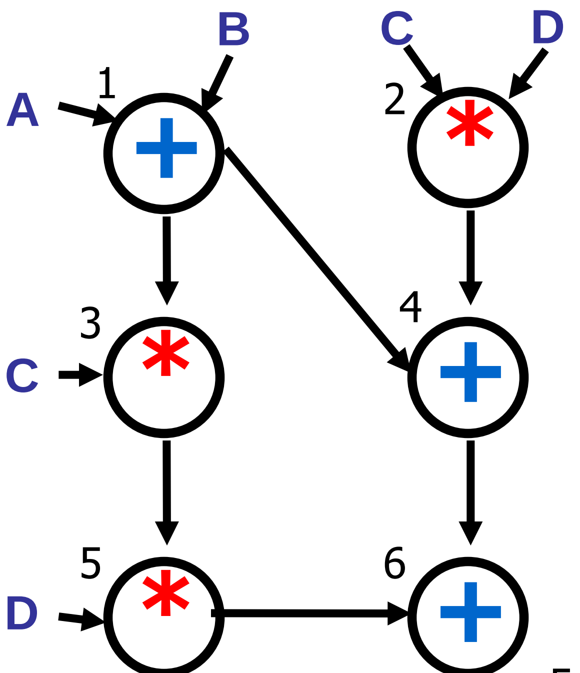
**NP-hard**
**Simulated annealing**
**Shiffted bottleneck**
**Branch-and-Bound**
**Gentic Algorithms**

# Jobshop Scheduling in UPPAAL



# Priced Timed Automata

# Task Graph Scheduling – Revisited




**Compute :**

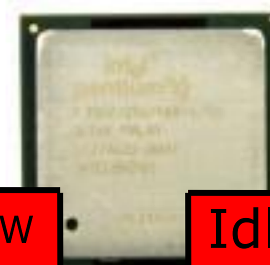
**(D * ( C * ( A + B )) + (( A + B ) + ( C * D ))**

**using 2 processors**

P1 (fast)


| | |
|---|---|
| + | 2ps |
| * | 3ps |
| Idle | 1oW |
| In use | 90W |

P2 (slow)

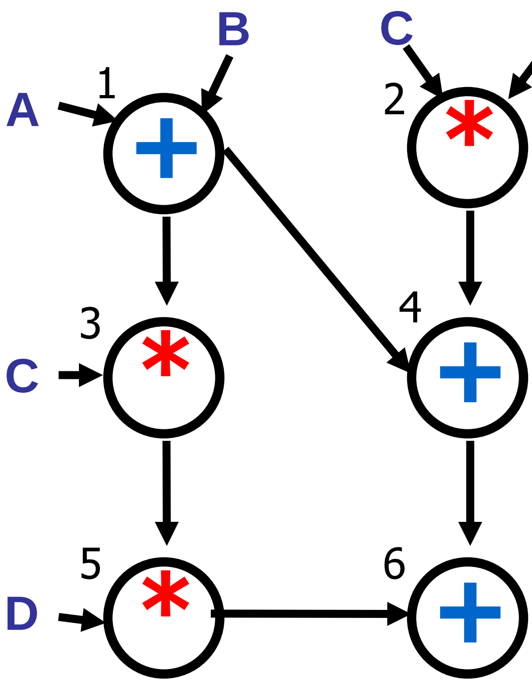


| | |
|---|---|
| + | 5ps |
| * | 7ps |
| Idle | 20W |
| In use | 30W |

**ENERGY:**


Energy:
1.39 nano-joule !!

# Task Graph Scheduling – Revisited




**Compute :**

**(D * ( C * ( A + B )) + (( A + B ) + ( C * D ))**

**using 2 processors**

P1 (fast)


| | |
|---|---|
| + | 2ps |
| * | 3ps |

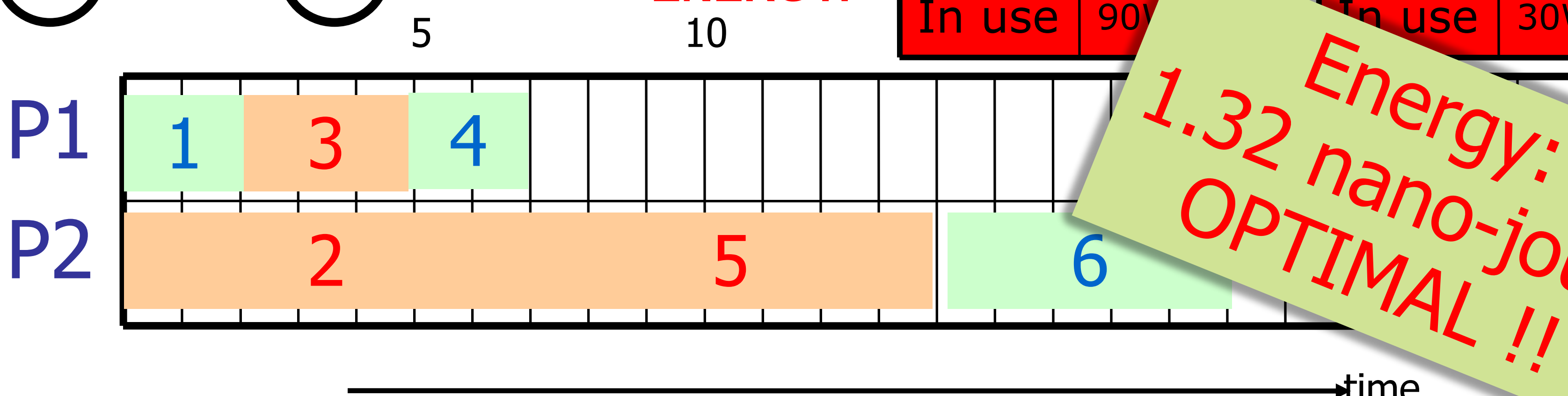
| | |
|---|---|
| Idle | 10W |
| In use | 90W |

P2 (slow)

| | |
|---|---|
| + | 5ps |
| * | 7ps |

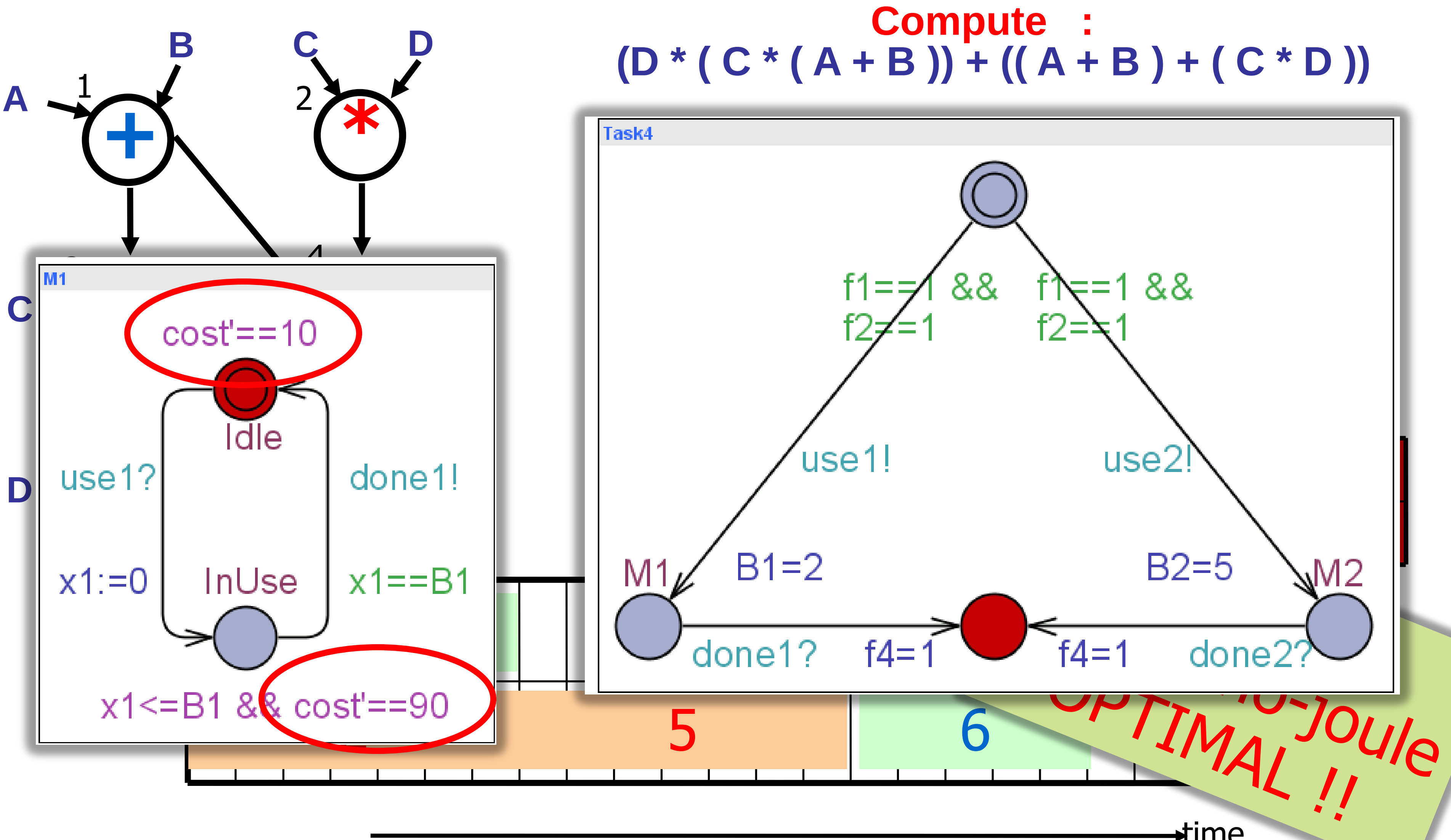
| | |
|---|---|
| Idle | 20W |
| In use | 30W |

**ENERGY:**



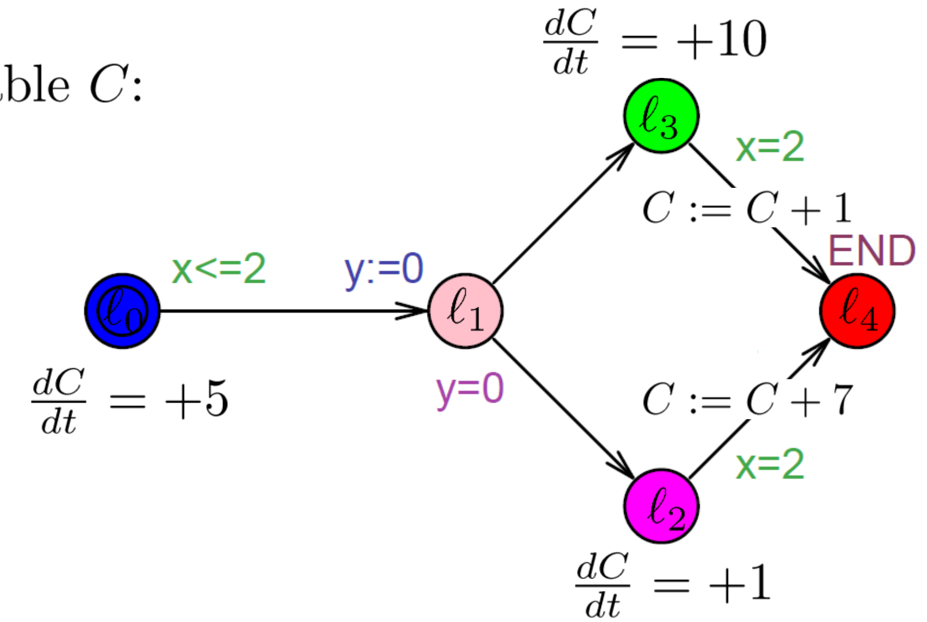

Energy:
1.32 nano-joule
OPTIMAL !!

# Task Graph Scheduling – Revisited



# A simple example

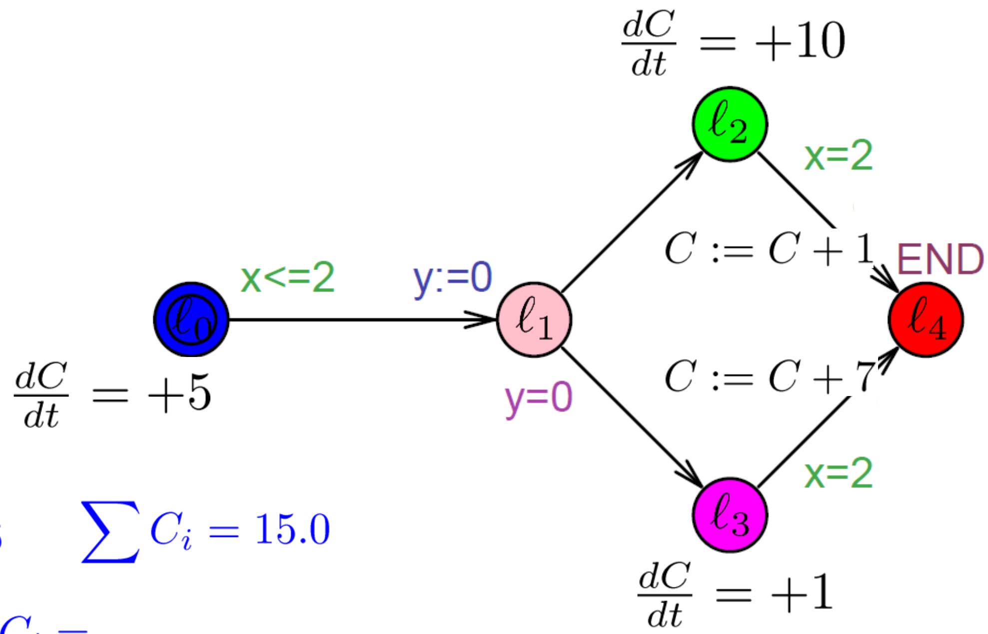
Observer variable $C$:




$$(\ell_0, [0,0]) \xrightarrow{1.9}_{9.5} (\ell_0, [1.9, 1.9]) \rightarrow_0 (\ell_1, [1.9, 0]) \rightarrow_0 (\ell_2, [1.9, 0]) \xrightarrow{0.1}_{0.1} (\ell_2, [2, 0.1]) \rightarrow_7 (\ell_4, [2, 0.1]) \qquad \sum C_i = 16.6$$

$$(\ell_0, [0,0]) \xrightarrow{1.2}_{6.0} (\ell_0, [1.2, 1.2]) \rightarrow_0 (\ell_1, [1.2, 0]) \rightarrow_0 (\ell_3, [1.2, 0]) \xrightarrow{0.8}_{8.0} (\ell_3, [2, 0.8]) \rightarrow_1 (\ell_4, [2, 0.8]) \qquad \sum C_i = 15.0$$

# A simple example




$\sum C_i = 16.6$ $\quad \sum C_i = 15.0$

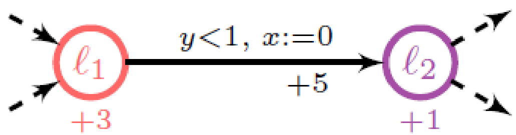
$\sum C_i = ..$

**Q**: What is cheapest cost for reaching $\ell_4$ ?

$$\inf_{0 \le t \le 2} \min\{5t + 10(2-t) + 1, 5t + (2-t) + 4\} = 9$$

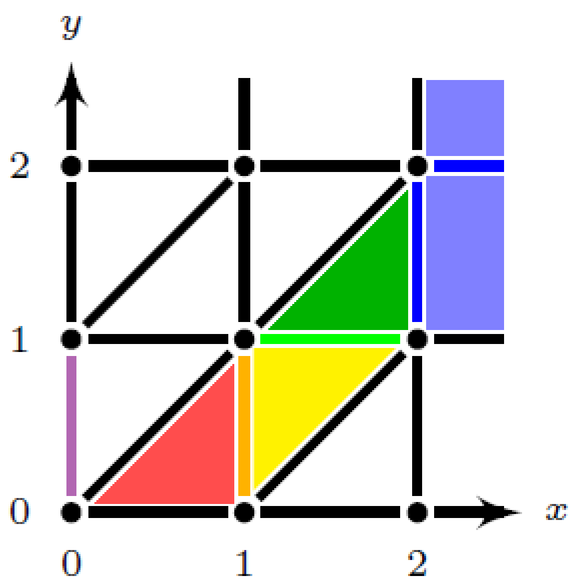
➜ **strategy:** leave immediately $\ell_0$, go to $\ell_3$, and wait there 2 t.u.

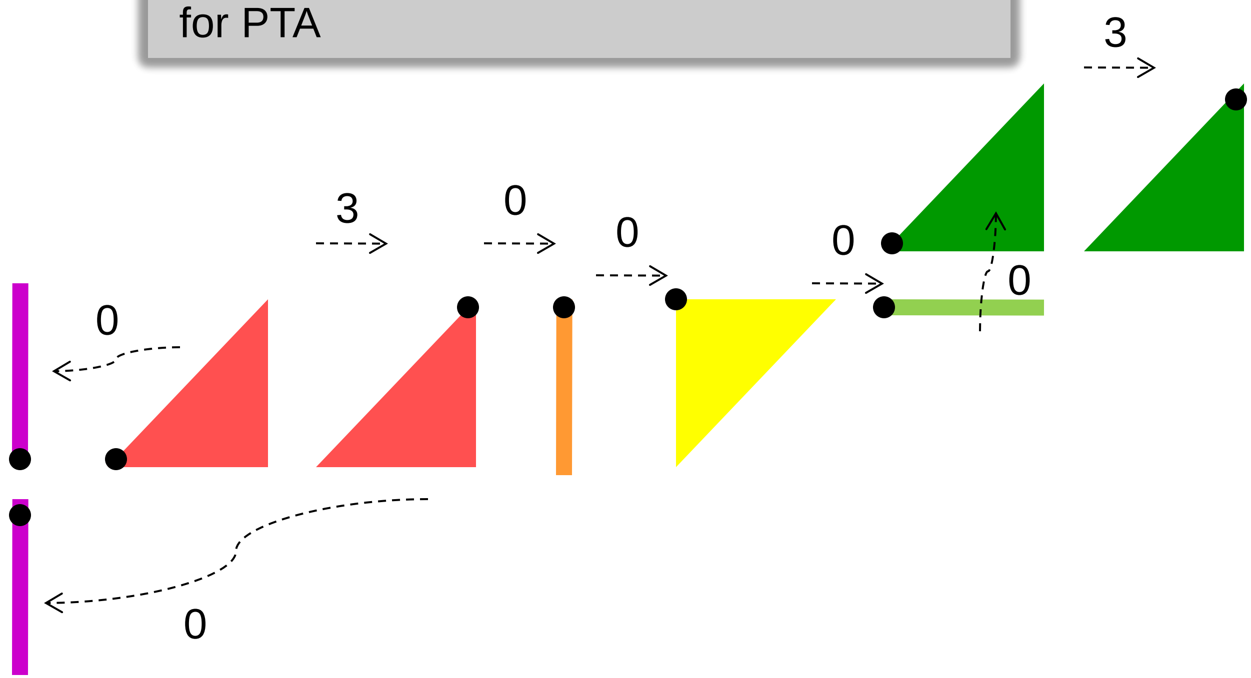
# Corner Point Regions

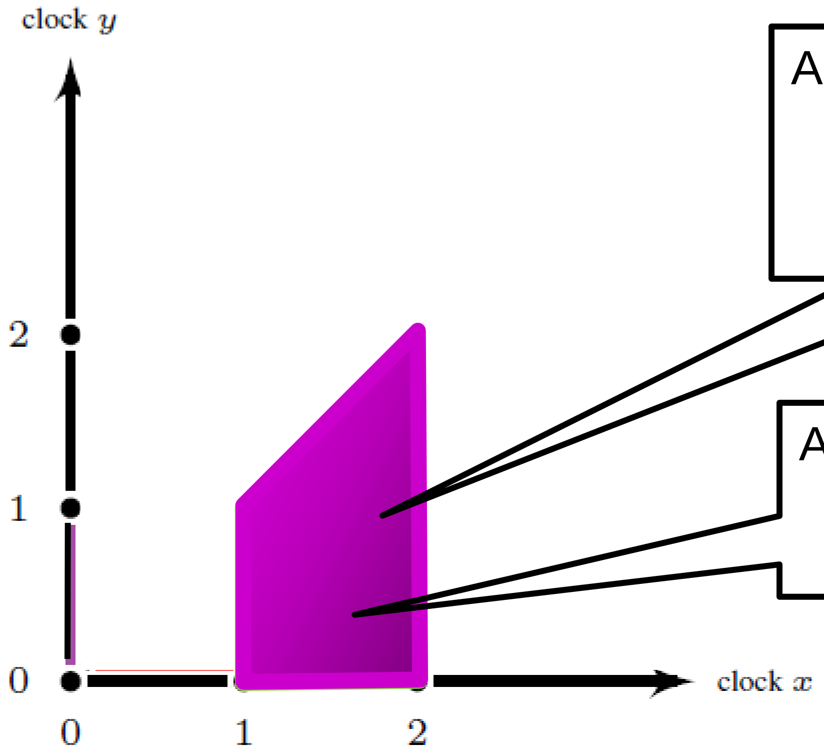



**THM** [Behrmann, Fehnker ..01] [Alur,Torre,Pappas 01]
Optimal reachability is decidable for PTA

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





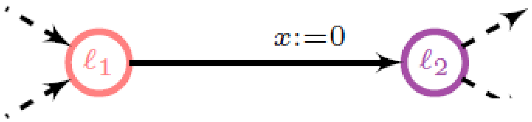

A zone **Z**:

$1 \leq x \leq 2 \quad \wedge$
$0 \leq y \leq 2 \quad \wedge$
$x - y \geq 0$

A cost function **C**

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

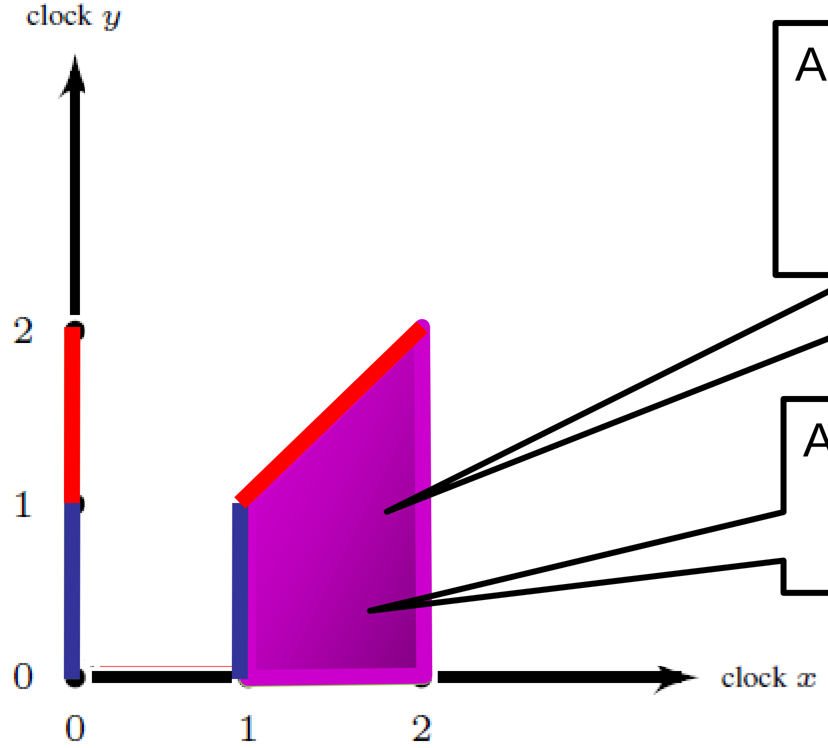
# Priced Zones – Reset [CAV01]




Z[x=0]:
x=0 $\wedge$
$0 \leq y \leq 2$

C = 1·y + 3

C= -1·y + 5




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

A cost function C
C(x,y) =
2·x - 1·y + 3

# Symbolic Branch & Bound Algorithm

**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

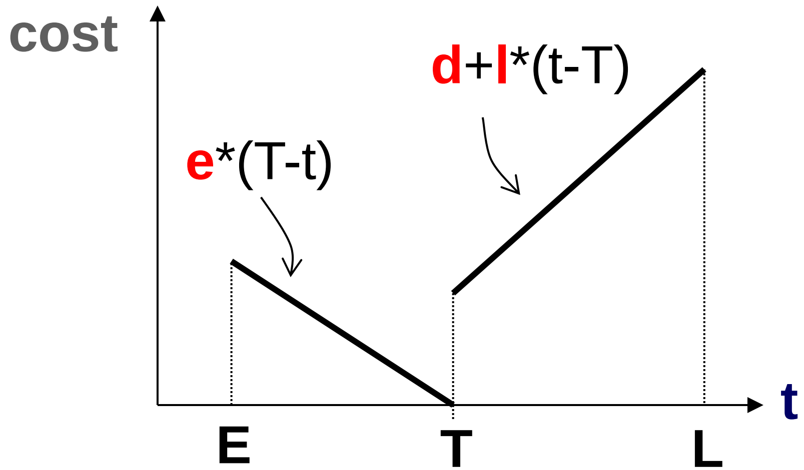
Cost := $\infty$
Passed := $\emptyset$
Waiting := $\{(l_0, Z_0)\}$
**while** Waiting $\neq \emptyset$ **do**
  **select** $(l, Z)$ from Waiting
  **if** $l = l_g$ and minCost$(Z)$ < Cost **then**
    Cost := minCost$(Z)$
  **if** minCost$(Z)$ + Rem$_{(l,Z)} \geq$ Cost **then** break
  **if** for all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

$Z' \leq Z$

***Z′* is bigger & cheaper than *Z***

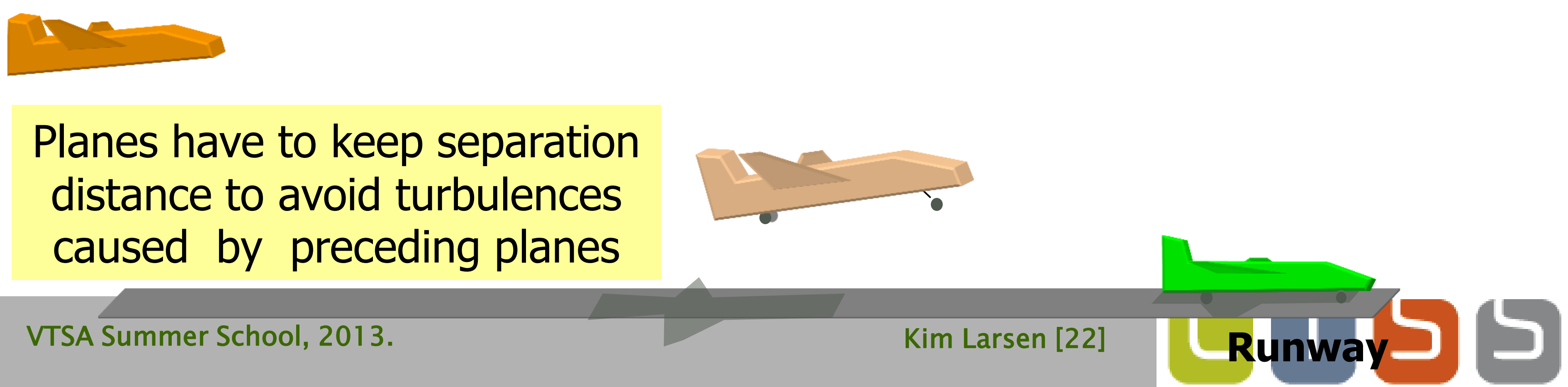
$\leq$ **is a well-quasi ordering which guarantees termination!**

# Example: Aircraft Landing

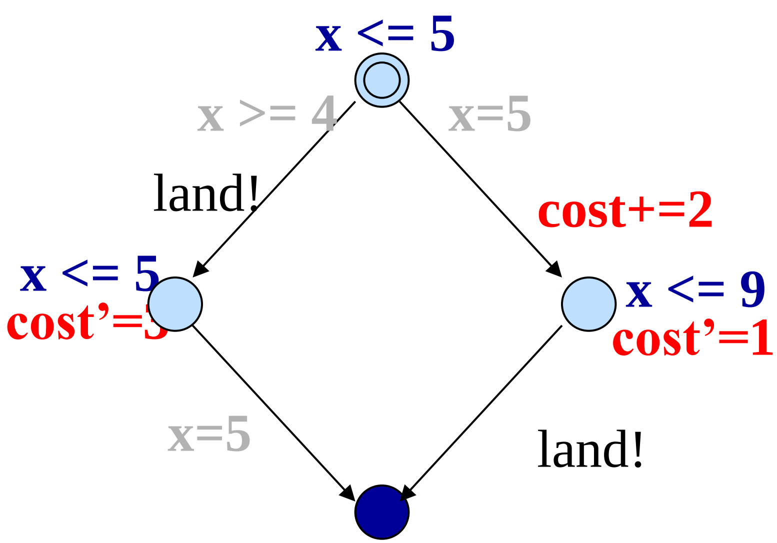



**E** earliest landing time
**T** target time
**L** latest time
**e** cost rate for being early
**l** cost rate for being late
**d** fixed cost for being late

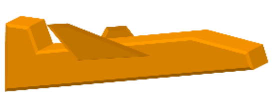
Planes have to keep separation distance to avoid turbulences caused by preceding planes



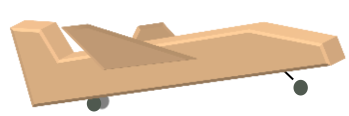
# Example: Aircraft Landing

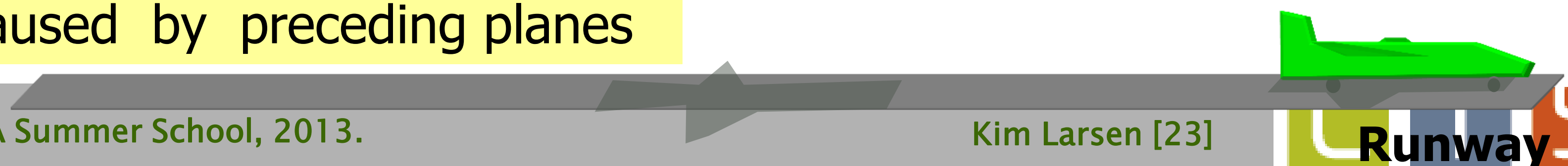



**4** earliest landing time
**5** target time
**9** latest time
**3** cost rate for being early
**1** cost rate for being late
**2** fixed cost for being late



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



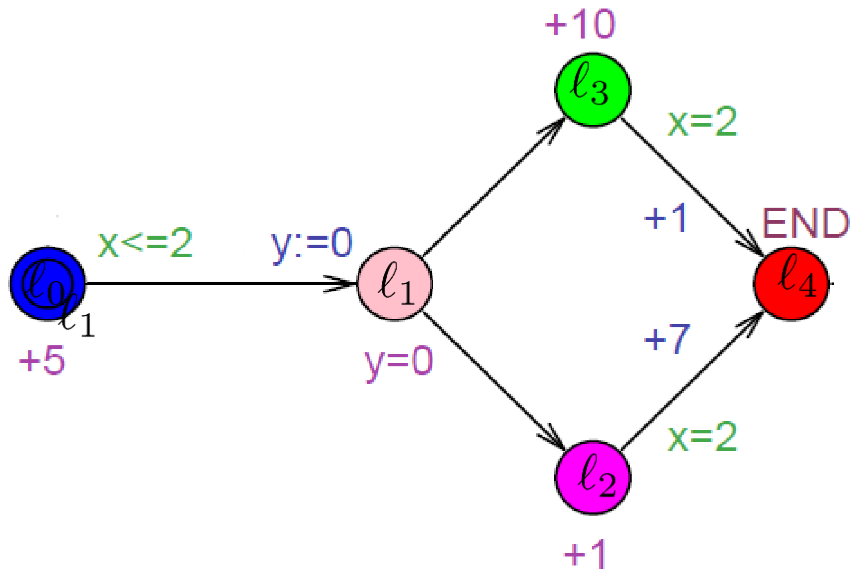


# Aircraft Landing

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

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

# Optimal Schedule



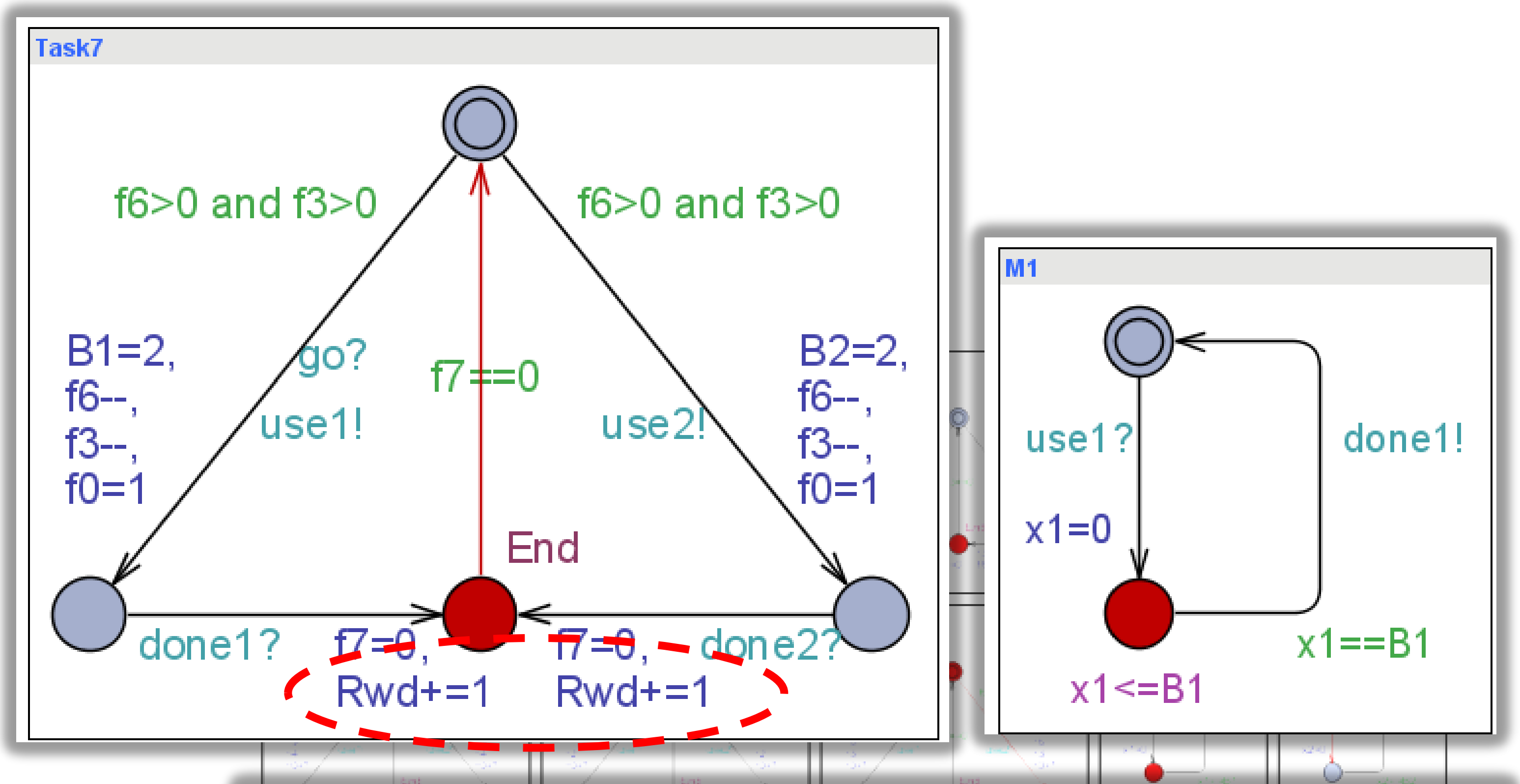

$$(\ell_0,[0,0]) \xrightarrow{1.2}_{6.0} (\ell_0,[1.2,1.2]) \rightarrow_0 (\ell_1,[1.2,0]) \rightarrow_0$$

$$(\ell_3,[1.2,0]) \xrightarrow{0.8}_{8.0} (\ell_3,[2,0.8]) \rightarrow_1 (\ell_4,[2,0.8])$$

$$\rightarrow_{2.0} (\ell_0,[0,0])$$

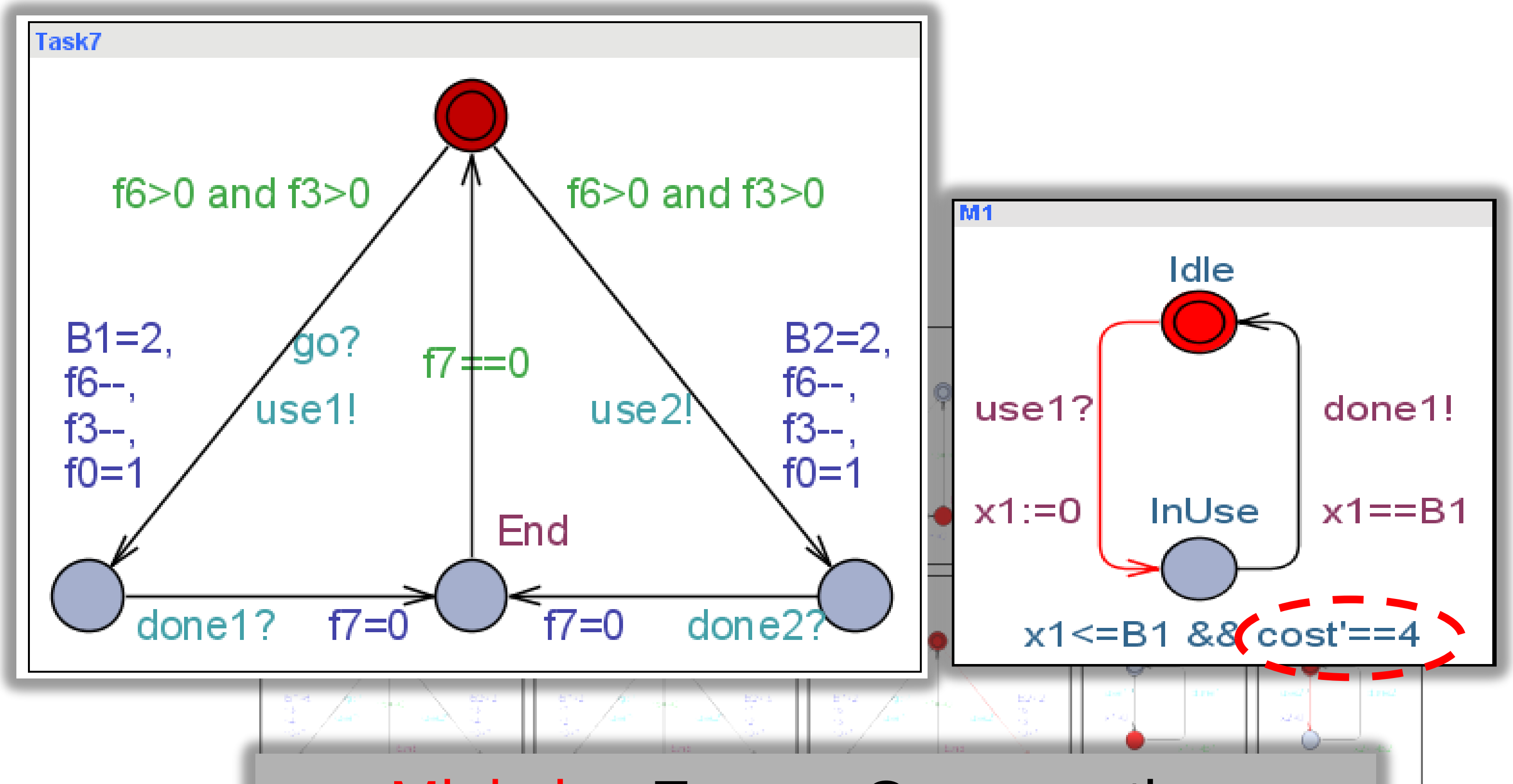
$$\boxed{\sum_i C_i / \sum_i t_i = 17/2 = 8.5}$$

# Optimal Infinite Scheduling



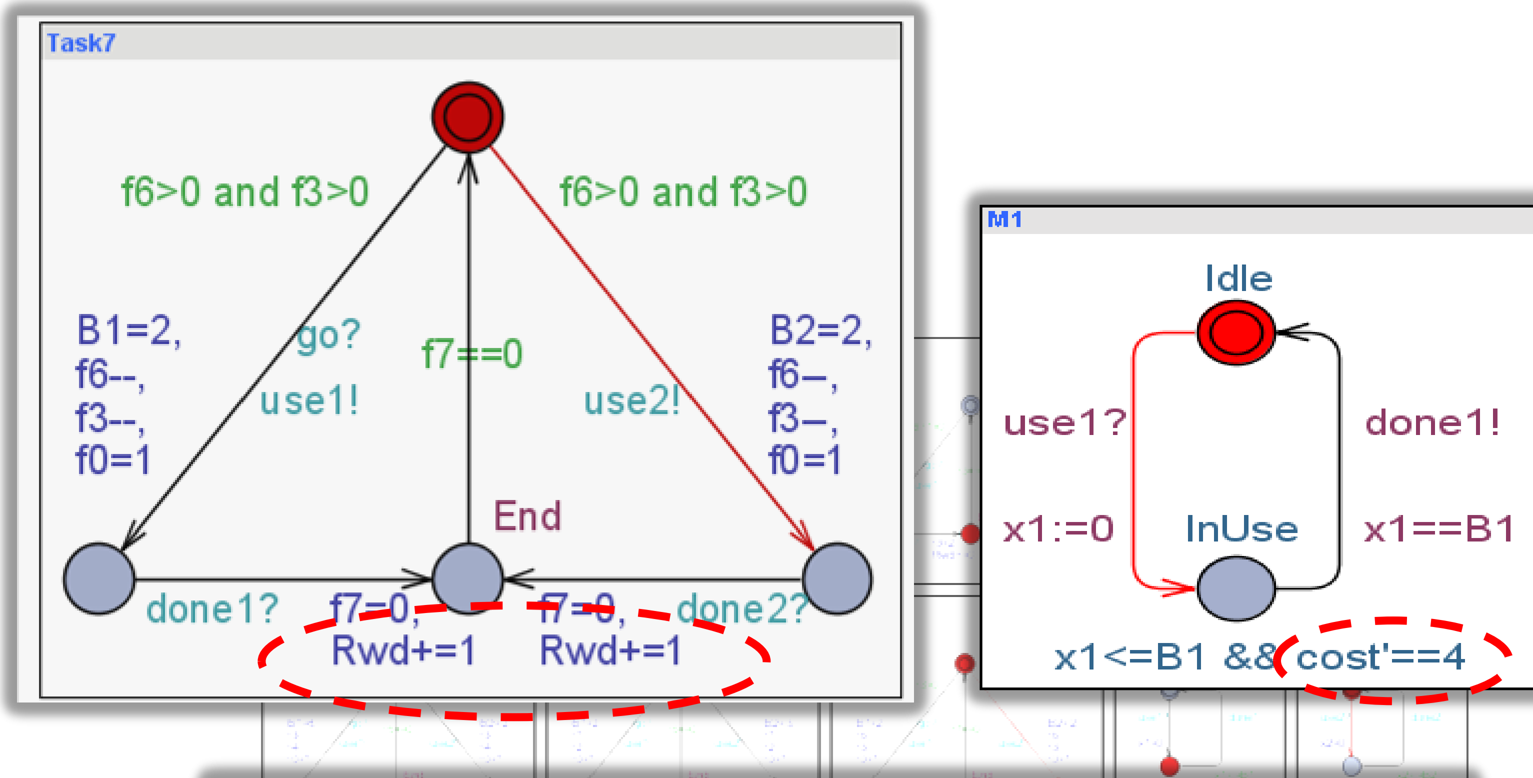

Maximize throughput:
i.e. maximize Reward / Time in the long run!

# Optimal Infinite Scheduling




Minimize Energy Consumption:
i.e. minimize Cost / Time in the long run

# Optimal Infinite Scheduling

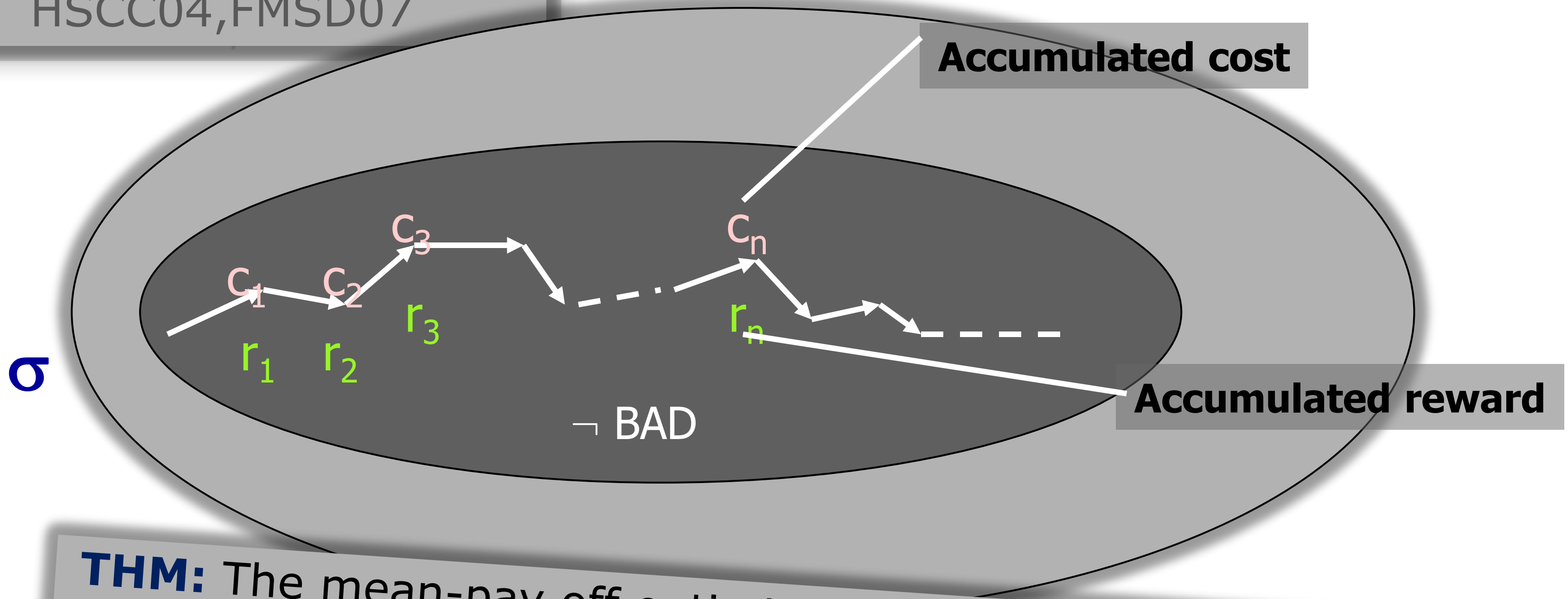



Maximize throughput:

i.e. maximize Reward / Cost in the long run

# Mean Pay-Off Optimality

Bouyer, Brinksma, Larsen:
HSCC04,FMSD07



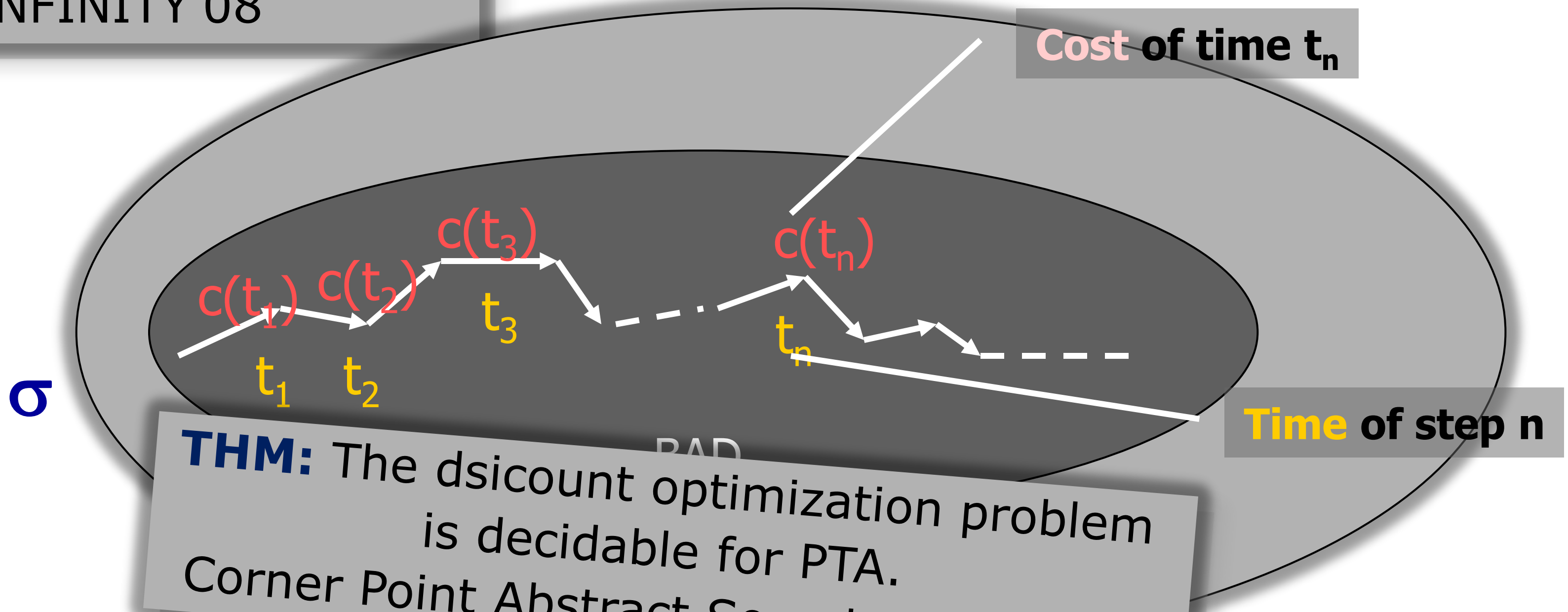

**THM:** The mean-pay off optimization problem is decidable (and PSPACE-complete) for PTA.
Corner Point Abstract Sound & Complete

Optimal Schedule $\sigma^*$: $val(\sigma^*) = \inf_\sigma val(\sigma)$

# Discount Optimality

$\lambda < 1$ : discounting factor

Larsen, Fahrenberg:
INFINITY'08




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

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

Optimal Schedule σ*: $\mathrm{val}(\sigma^*) = \inf_\sigma \mathrm{val}(\sigma)$

# 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:

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

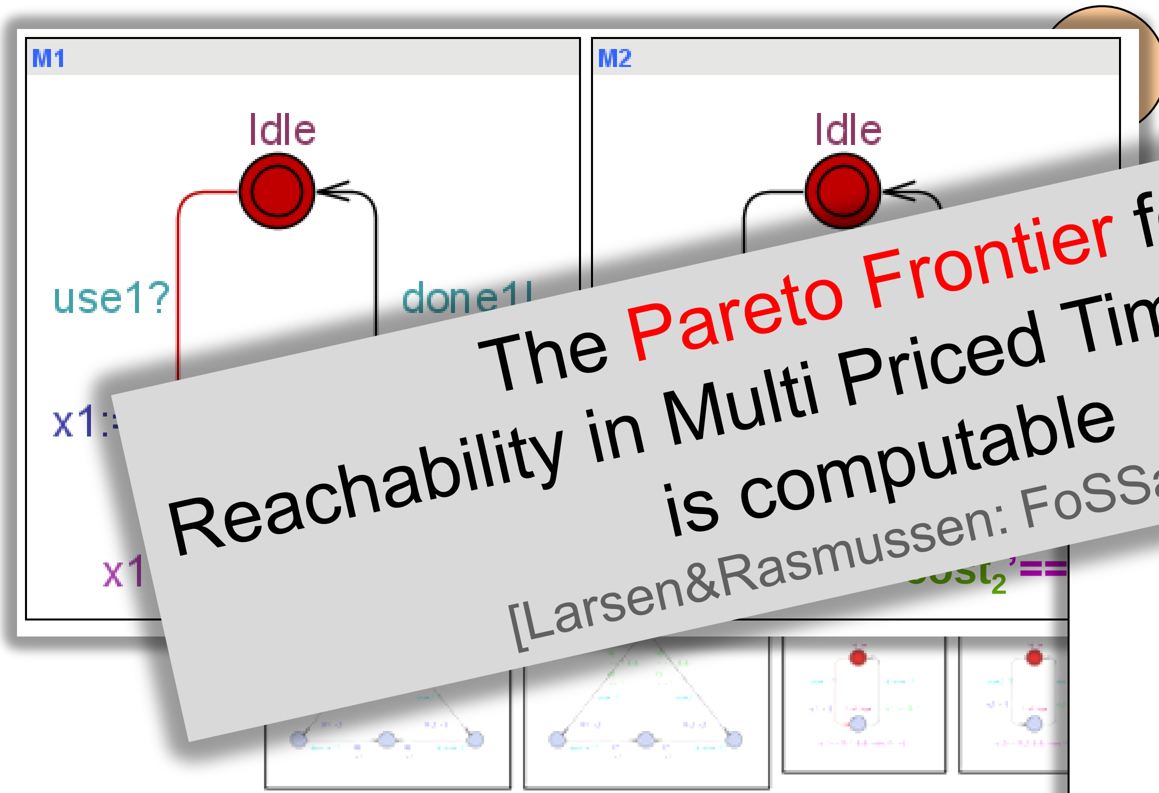
then $\inf_Z f$ is obtained at a corner-point of $Z$ (with integer coefficients).

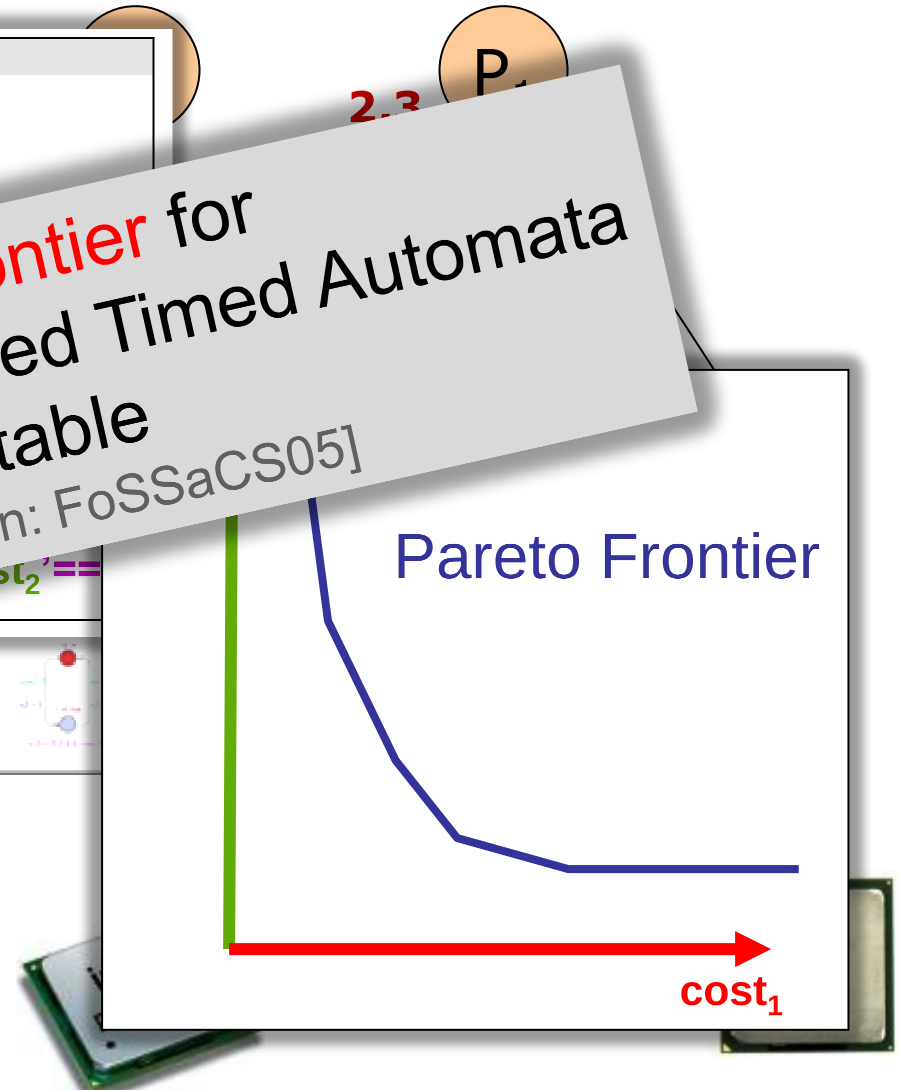
**Lemma**
Let $Z$ be a (bounded, closed) zone and let $f$ be a function over $Z$ defined by:

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

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

# Multiple Objective Scheduling




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

[Larsen&Rasmussen: FoSSaCS05]

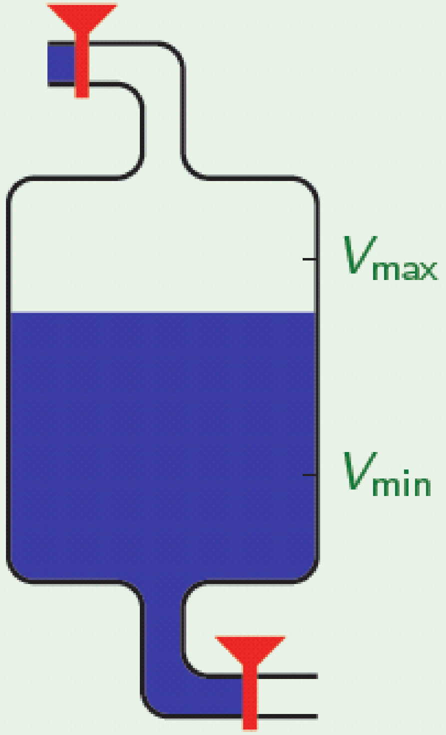
# Energy Automata

# Managing Resources

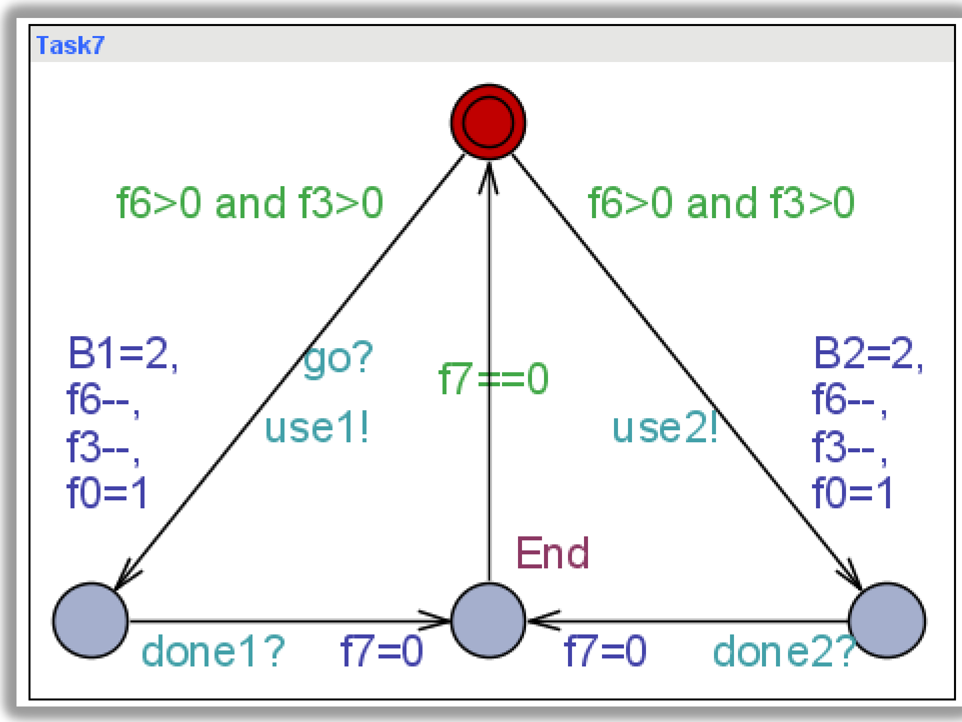
## Example

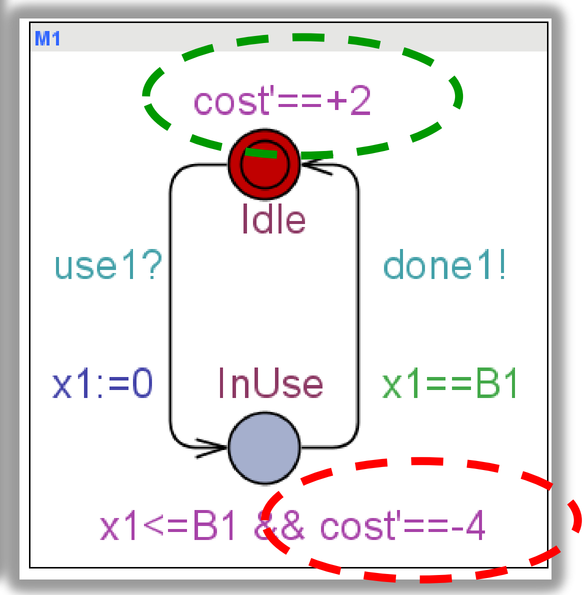
In some cases, resources can both be consumed and regained.

The aim is then to keep the level of resources within given bounds.



# Consuming & Harvesting Energy

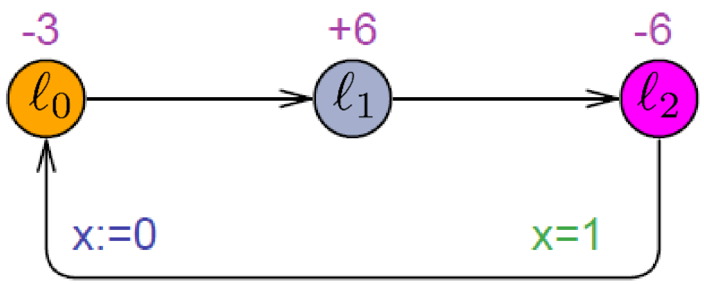


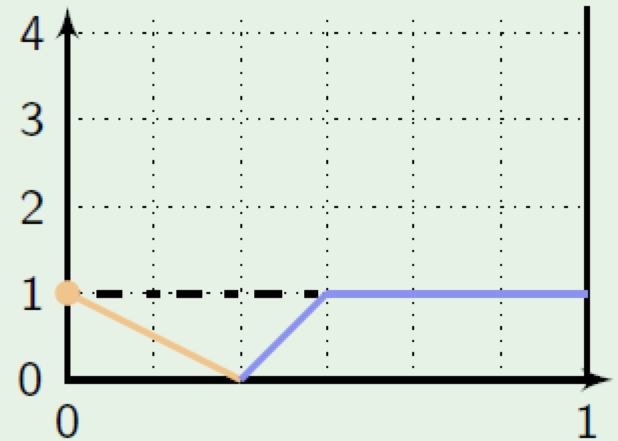




Maximize throughput
while respecting: $0 \leq E \leq MAX$

# Energy Constrains

- Energy is not only consumed but may also be regained
- The aim is to continously satisfy some energy constriants




lower-weak-upper-bound problem

# Results (so far)

Bouyer, Fahrenberg, Larsen, Markey, Srba: **FORMATS 2008**

## Untimed

| | games | existential problem | universal problem |
|---|---|---|---|
| L | $\in$ UP $\cap$ coUP<br>P-h | $\in$ P | $\in$ P |
| L+W | $\in$ NP $\cap$ coNP<br>P-h | $\in$ P | $\in$ P |
| L+U | EXPTIME-c | $\in$ PSPACE<br>NP-h | $\in$ P |

## 1 Clock

| | games | existential problem | universal problem |
|---|---|---|---|
| L | ? | $\in$ P | $\in$ P |
| L+W | ? | $\in$ P | $\in$ P |
| L+U | undecidable | ? | ? |

Corner Point Abstraction Suffice

Corner Point Abstraction Insufficient

## 1½ Clocks !!

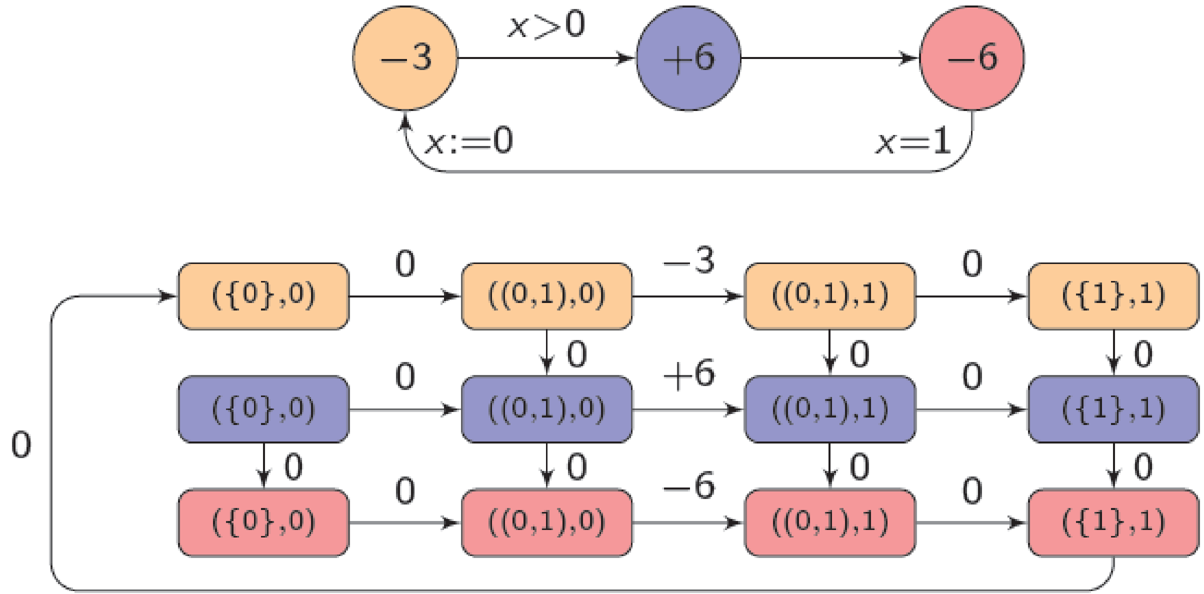
## 2+ Clocks ???

# L–Problem for 1–Clock Case

**Theorem:**
The L-problem is decidable in **PTIME** for **1**-clock PTAs

*Proof.*

- Corner-point abstraction:




P Bouyer, U Fahrenberg, K Larsen, N Markey,.. . Infinite runs in weighted timed automata with energy constraints. 2008.

# LU–Problem for 1–Clock Energy Games

**Theorem**
For 1-clock priced timed games, the existence of a strategy satisfying LU-bounds is **undecidable**

*Proof.*

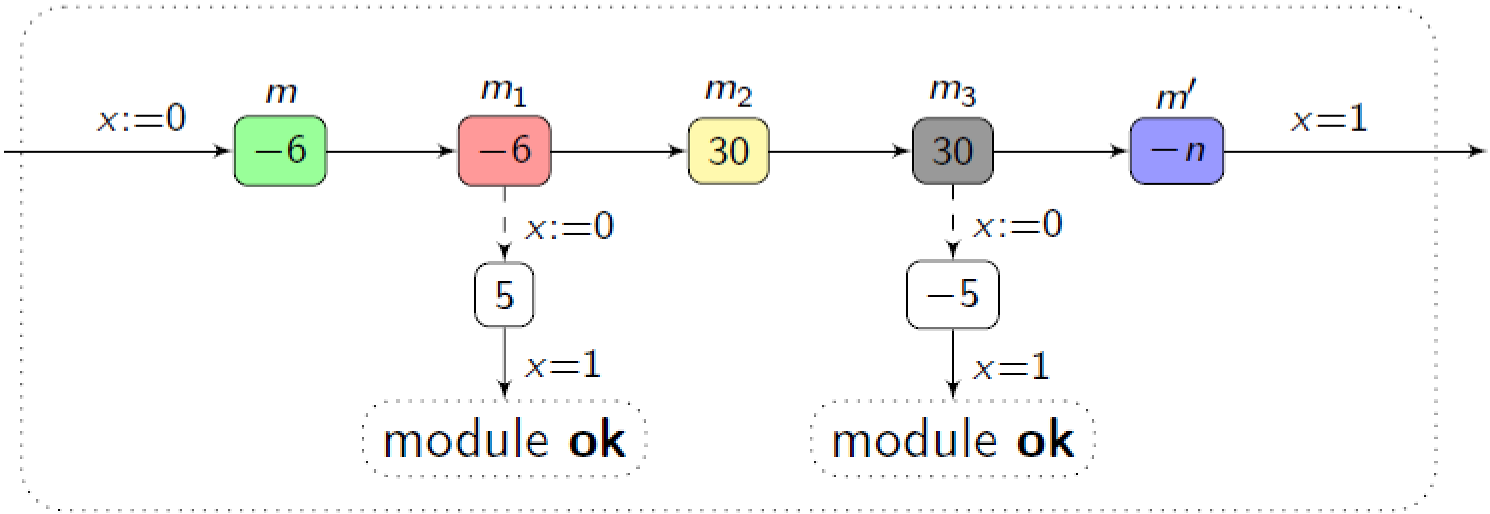
- we encode a 2-counter machine:
  - each instruction is encoded as a module;
  - the values $c_1$ and $c_2$ of the counters are encoded by energy level

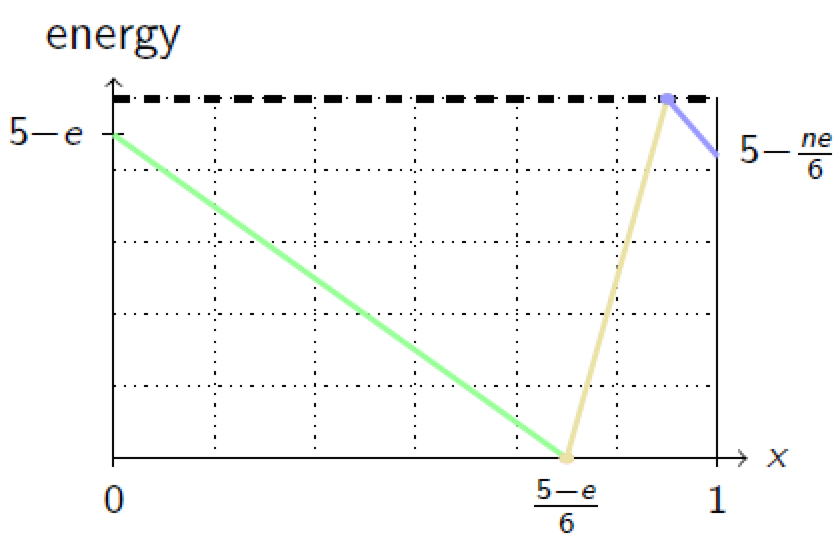
$$e = 5 - \frac{1}{2^{c_1} \cdot 3^{c_2}}$$

when entering the corresponding module.

P Bouyer, U Fahrenberg, K Larsen, N Markey,.. . Infinite runs in weighted timed automata with energy constraints. 2008.

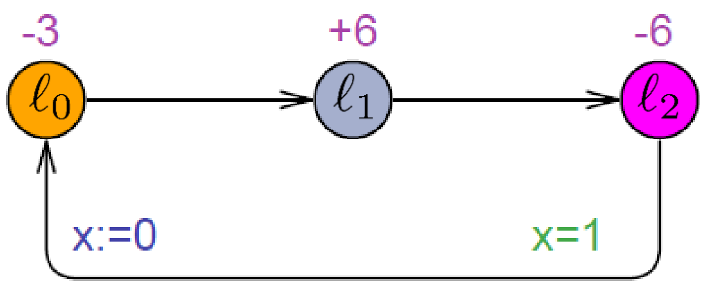
# Generic Module for Inc/Dec

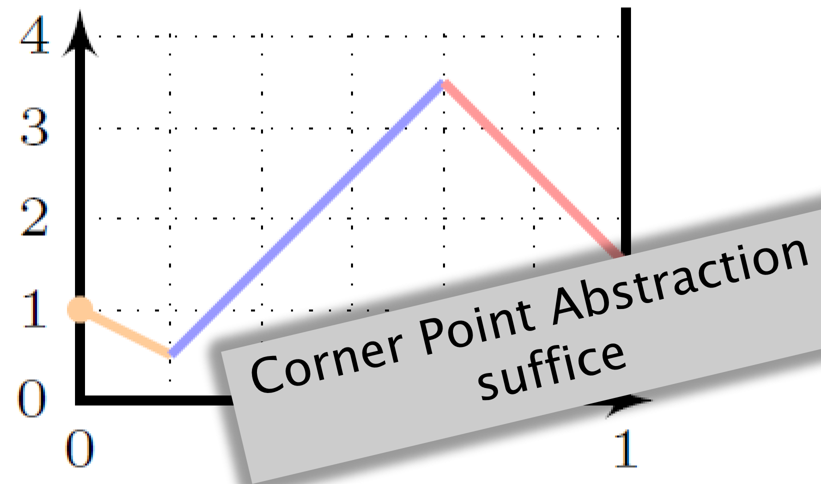


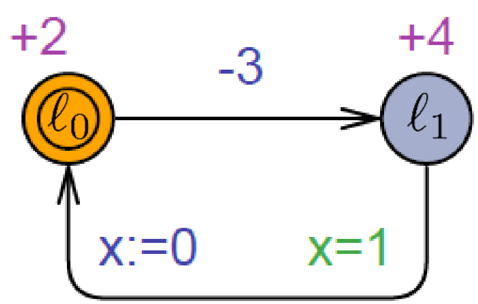



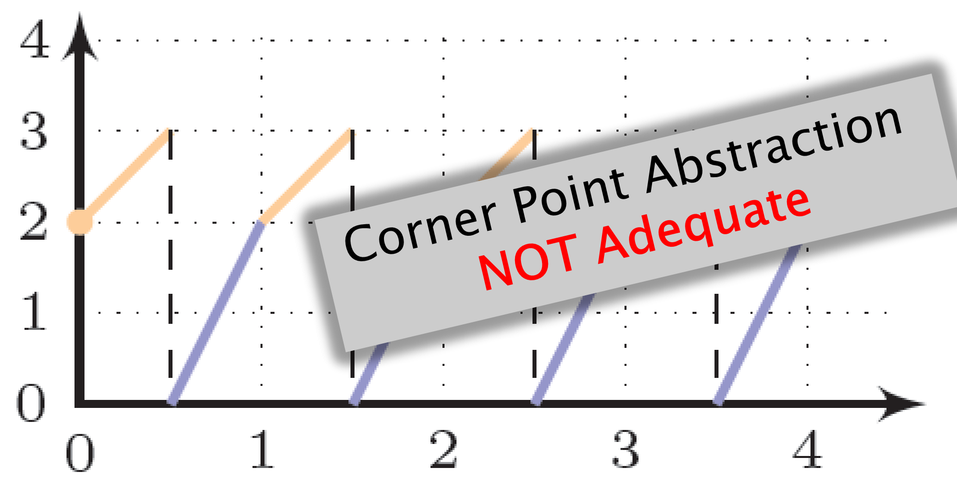

- $n = 3$: increment $c_1$
- $n = 2$: increment $c_2$
- $n = 12$: decrement $c_1$
- $n = 18$: decrement $c_2$

# 1½ Clocks = Discrete Updates

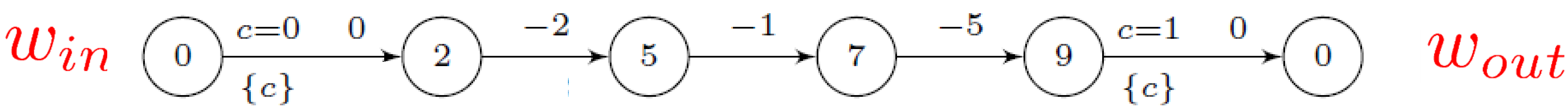


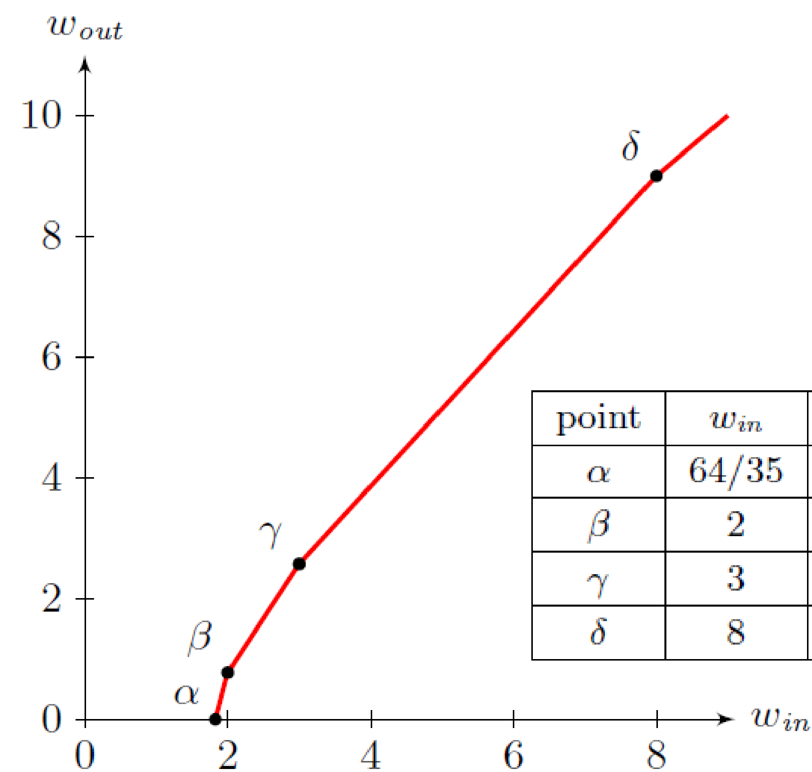



# New Approach: Energy Functions



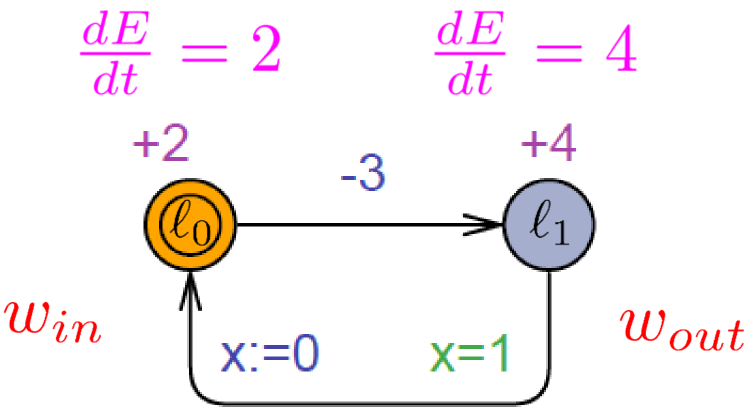

- Maximize energy along paths
- Use this information to solve general problem



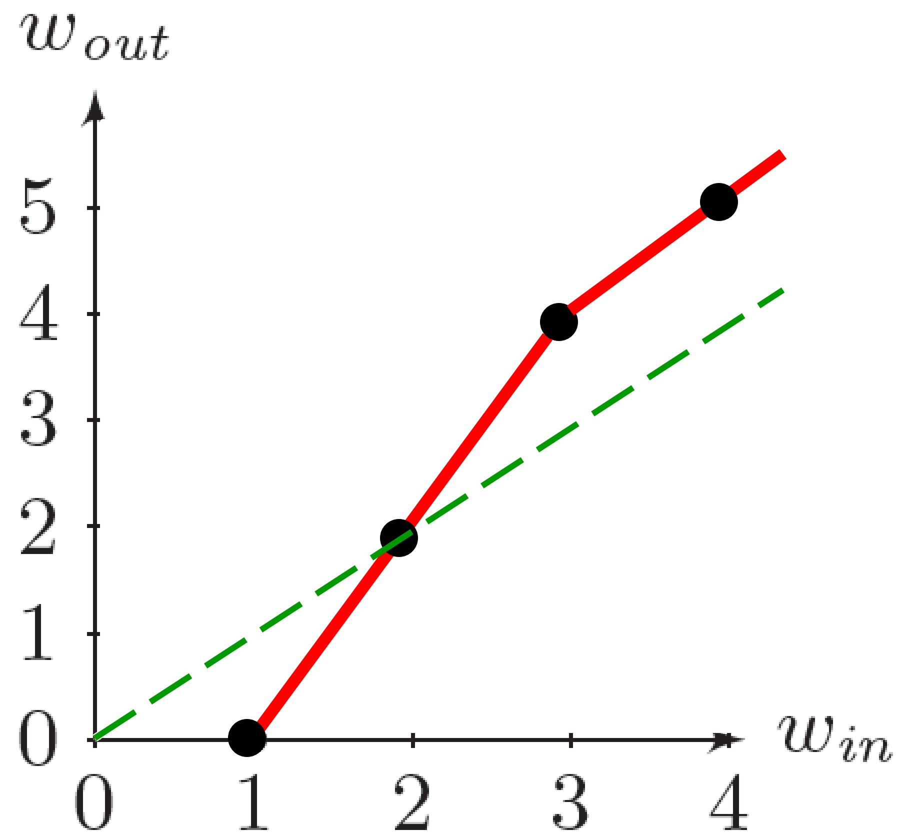

| point | $w_{in}$ | $w_{out}$ |
|---|---|---|
| $\alpha$ | 64/35 | 0 |
| $\beta$ | 2 | 27/35 |
| $\gamma$ | 3 | 18/7 |
| $\delta$ | 8 | 9 |

P. Bouyer, U. Fahrenberg, K. G. Larsen, N. Markey: Timed automata with observers under energy constraints. HSCC 2010

# Energy Function



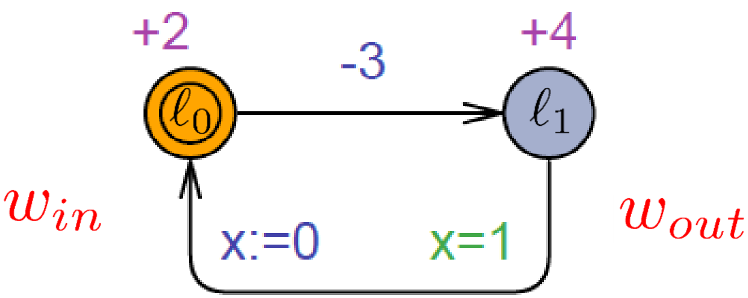

**General Strategy**
Spend just enough time to survive the next negative update

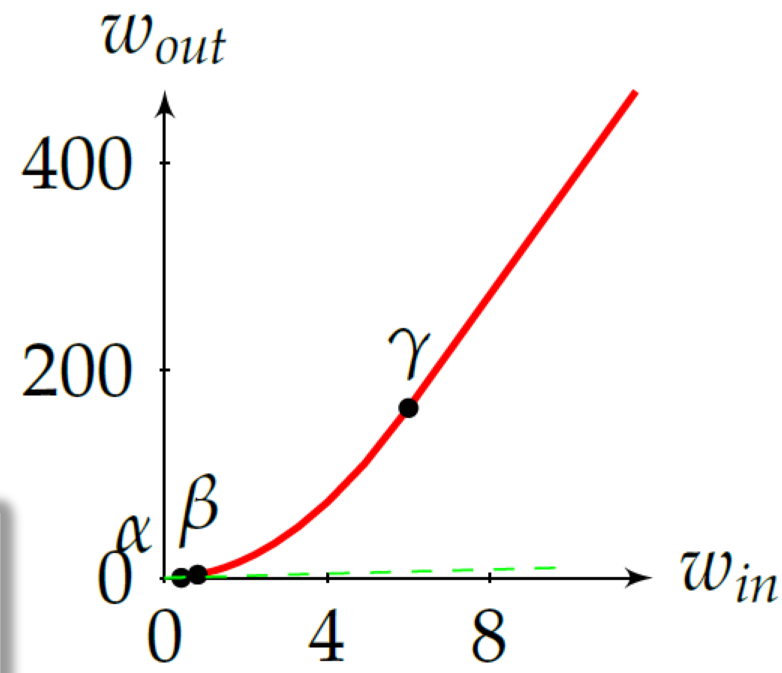



P. Bouyer, U. Fahrenberg, K. G. Larsen, N. Markey: Timed automata with observers under energy constraints. HSCC 2010

# Exponential PTA

$\frac{dE}{dt} = 2E$ $\qquad$ $\frac{dE}{dt} = 4E$




**General Strategy**

Spend just enough time

~~to survive the next negative update~~

so that after next negative update there is a certain positive amount !

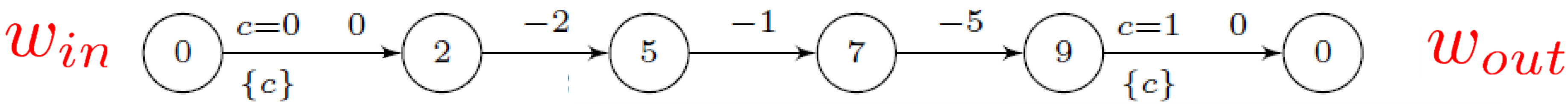
**Minimal Fixpoint:**

$$\frac{3}{e^2-1} \approx 0.47$$

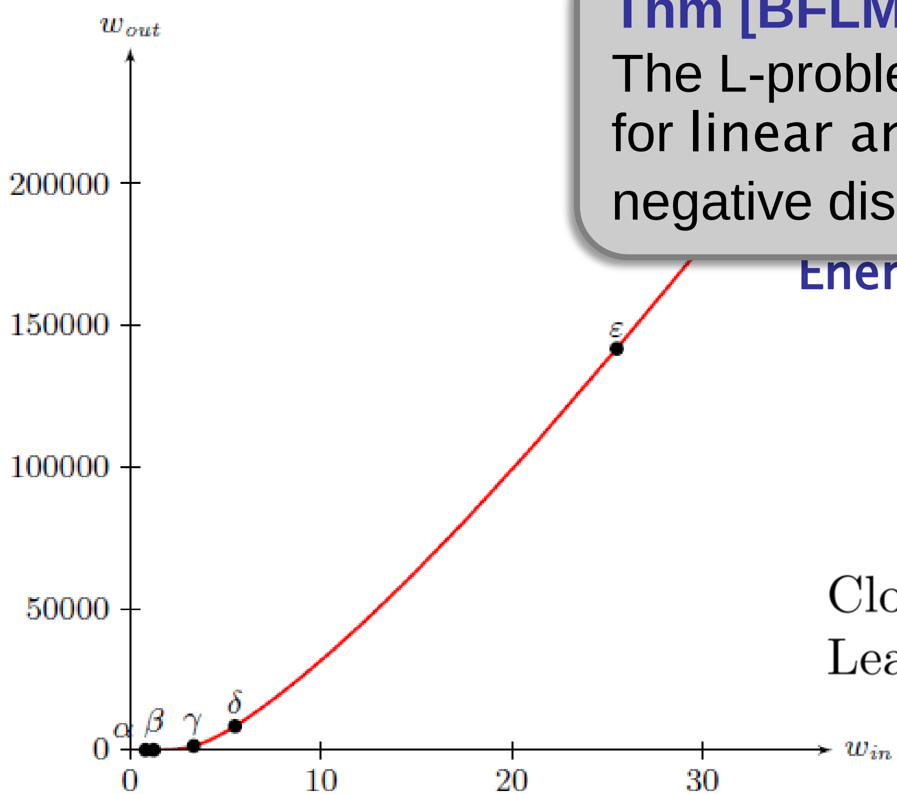
P. Bouyer, U. Fahrenberg, K. G. Larsen, N. Markey: Timed automata with observers under energy constraints. HSCC 2010

# Exponential PTA

$$\frac{dE}{dt} = 2E$$




**Thm [BFLM09]:**
The L-problem is **decidable**
for linear and exponential 1-clock PTAs with negative discrete updates.




### Energy Function

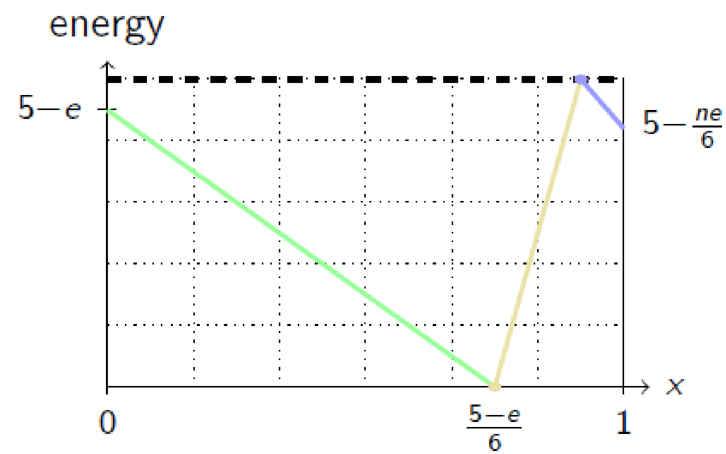
- $f : x \mapsto \alpha \cdot x^r + \beta$ where $r$ is rational
- $\frac{df}{dt} \geq 1$

Closed under max and composition.
Least fixed point computable.

P. Bouyer, U. Fahrenberg, K. G. Larsen, N. Markey: Timed automata with observers under energy constraints. HSCC 2010

# Multiple Costs & Clocks

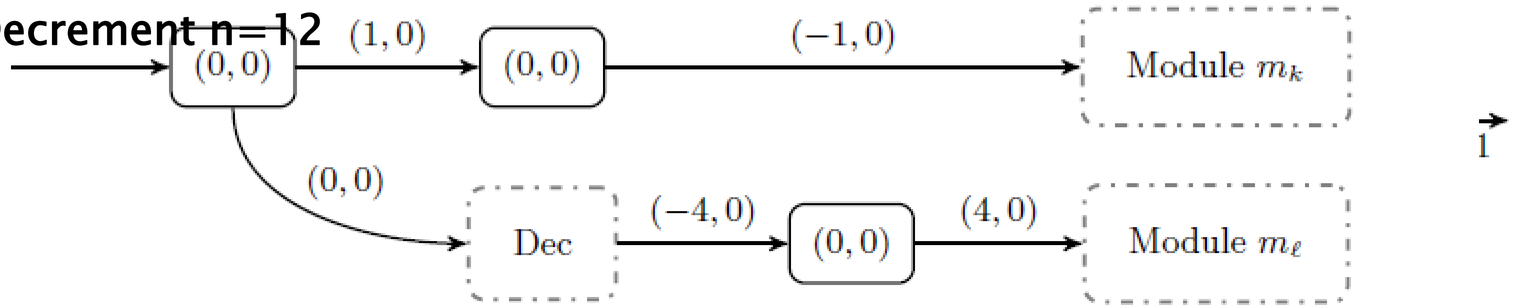
- LU Problem **Undecidable**
  - 2 clocks + 2 costs [1]
  - 1 clock + 2 costs [2]
  - 2 clocks + 1 cost [3]




**Test-Decrement**

Increment n=3

Decrement n=12

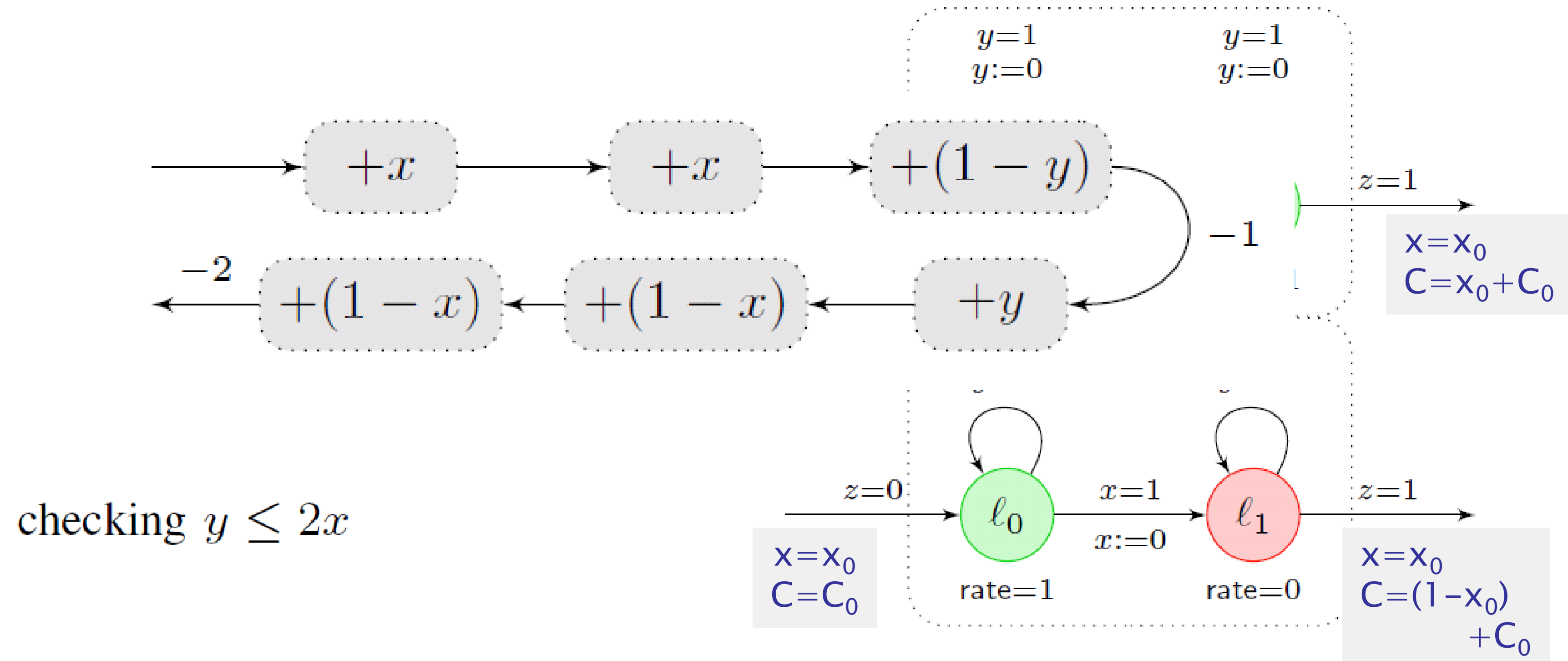




(1) Karin Quaas. On the interval-bound problem for weighted timed automata. 2011.
(2) Uli Fahrenberg, Line Juhl, Kim G. Larsen, and Jiri Srba. Energy games in multiweighted automata. 2011
(3) Nicolas Markey. Verification of Embedded Systems – Algorithms and Complexity. 2011.

# Multiple Clocks & 1 Cost

- L problem is **undecidable**
  - 4 clocks or more




P. Bouyer, K. G. Larsen, and N. Markey. Lower-bound constrained runs in weighted timed automata. QEST 2012

# Multiple Costs & 0 Clocks

| # weights | Bound | Existential | Universal | Game |
|---|---|---|---|---|
| One | L | $\in$ P [4] | $\in$ P [4] | $\in$ UP $\cap$ coUP [4] |
| | LW | $\in$ P [4] | $\in$ P [4] | $\in$ NP $\cap$ coNP [4] |
| | LU | NP-hard [4], $\in$ PSPACE [4] | $\in$ P [4] | EXPTIME-complete [4] |
| Fixed ($k > 1$) | L | **NP-hard**, $\in$ $k$-EXPTIME [3] (Remark 17) | $\in$ P (Remark 18) | **EXPTIME-hard**, $\in$ $k$-EXPTIME [3] (Remark 19) |
| | LW | **NP-hard**, $\in$ **PSPACE** **PSPACE-complete** for $k \geq 4$ (Remark 20) | $\in$ P (Remark 18) | **EXPTIME-complete** (Remark 21) |
| | LU | **PSPACE-complete** (Remark 20) | $\in$ P (Remark 18) | **EXPTIME-complete** (Remark 21) |
| Arbitrary | L | **EXPSPACE-complete** (Theorem 9) | $\in$ P (Remark 18) | **EXPSPACE-hard** (from EL) decidable [3] |
| | LW | **PSPACE-complete** (Theorem 9) | $\in$ P (Remark 18) | **EXPTIME-complete** (Remark 21) |
| | LU | **PSPACE-complete** (Theorem 9) | $\in$ P (Remark 18) | **EXPTIME-complete** (Remark 21) |

Uli Fahrenberg, Line Juhl, Kim G. Larsen, and Jirı Srba. Energy games in multiweighted automata. 2011

# Conclusion

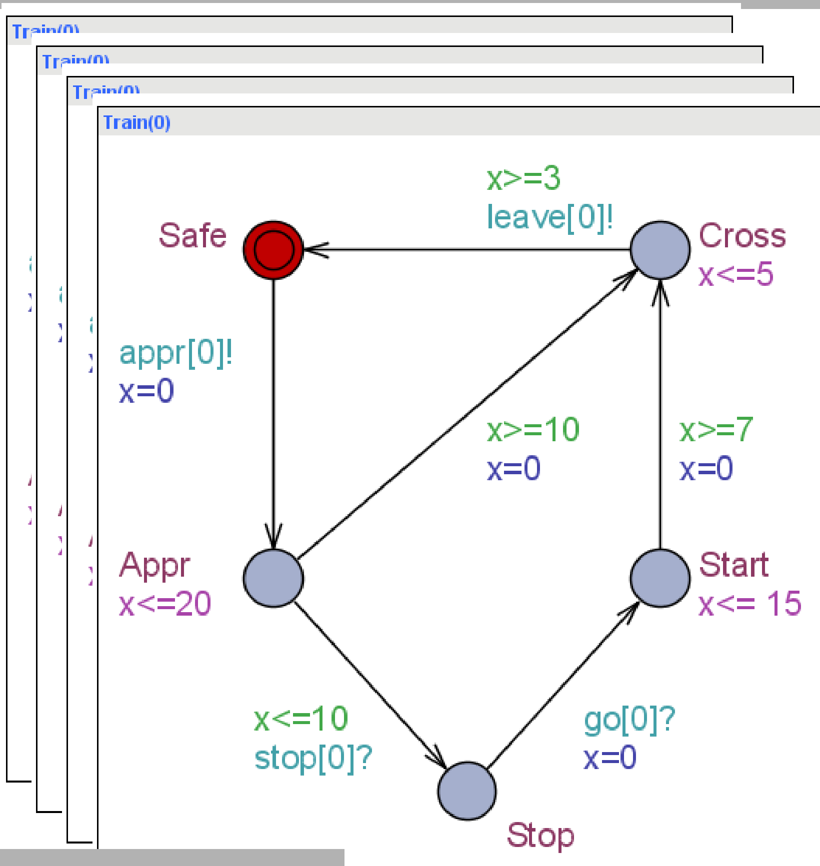
- Priced Timed Automata a uniform framework for modeling and solving dynamic ressource allocation problems!
- Not mentioned here:
  - Model Checking Issues (ext. of CTL and LTL).
- Future work:
  - Zone-based algorithm for optimal infinite runs.
  - Approximate solutions for priced timed games to circumvent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

# Timed Games

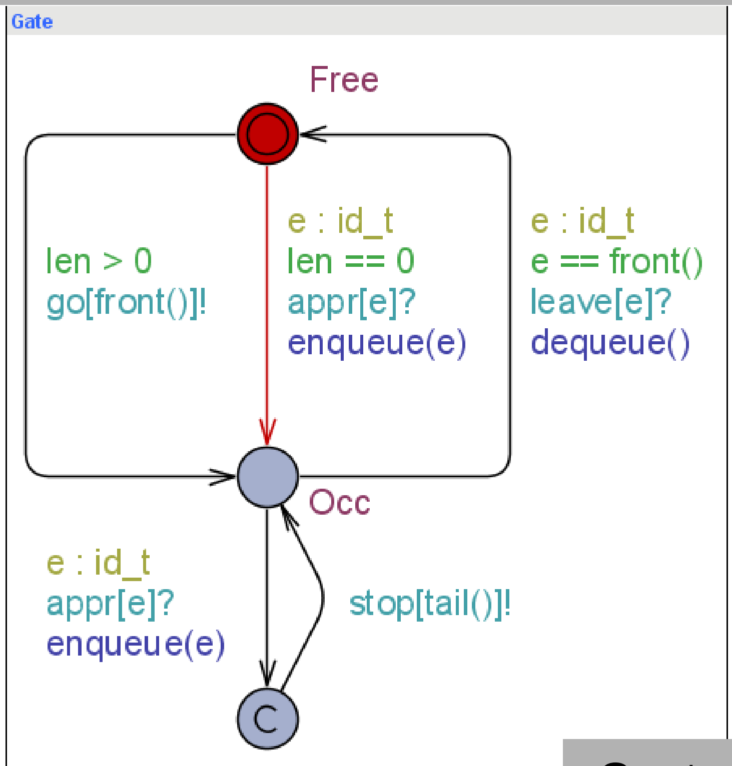
TIGA

Kim G. Larsen
CISS – Aalborg University
DENMARK

# Model Checking



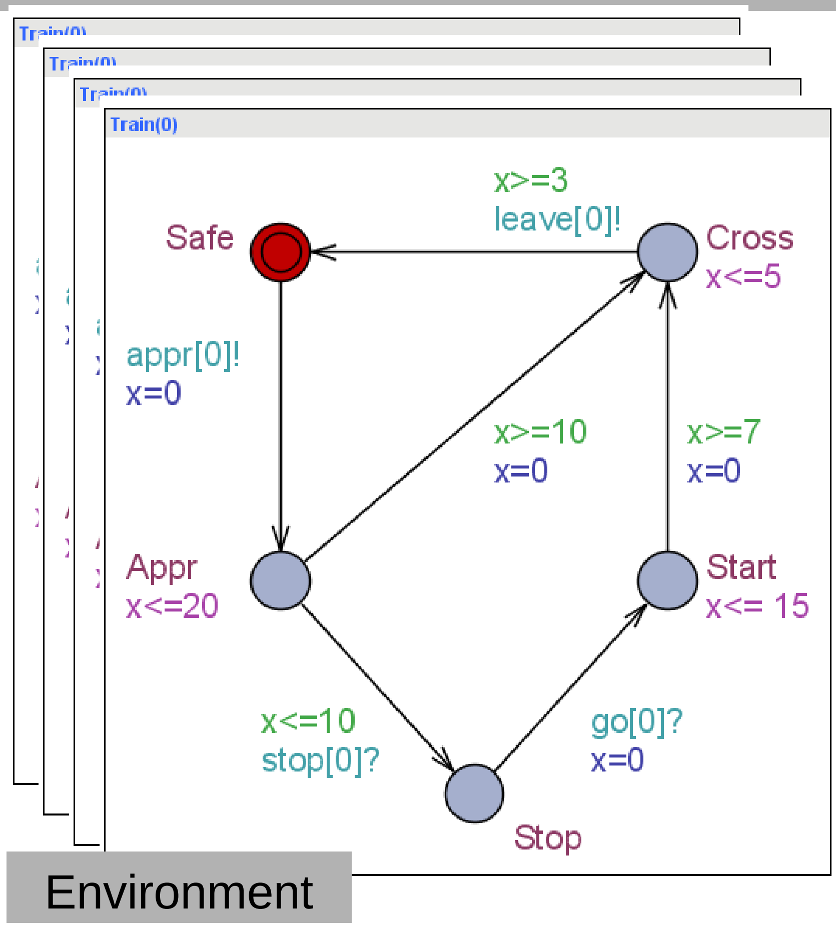

Environment

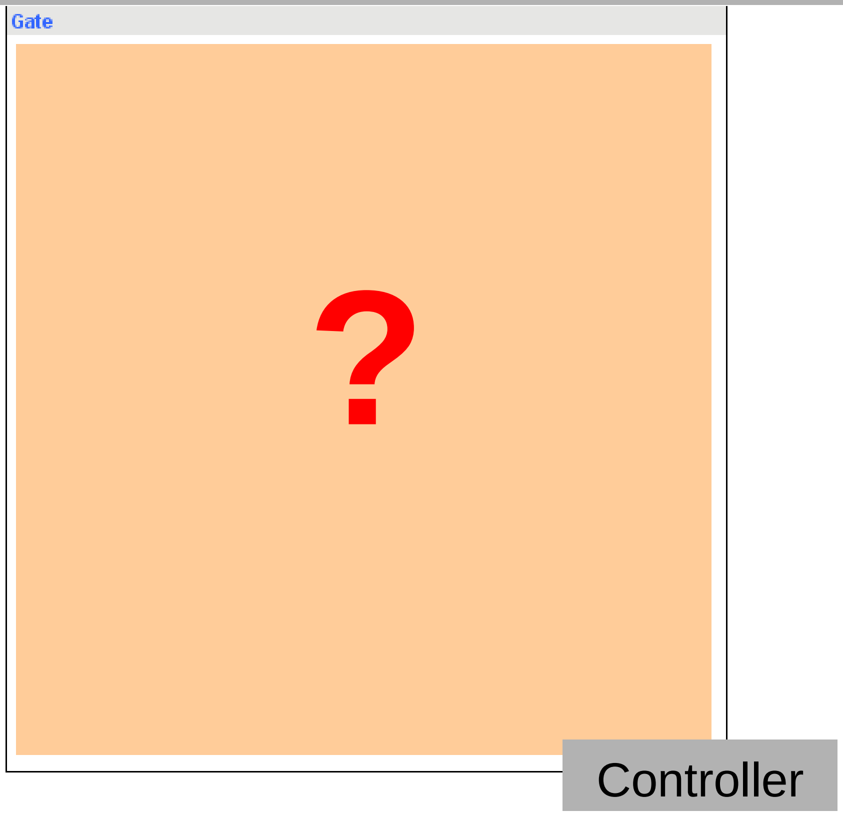



Controller

$\phi$: Never two trains at the crossing at the same time

# Synthesis



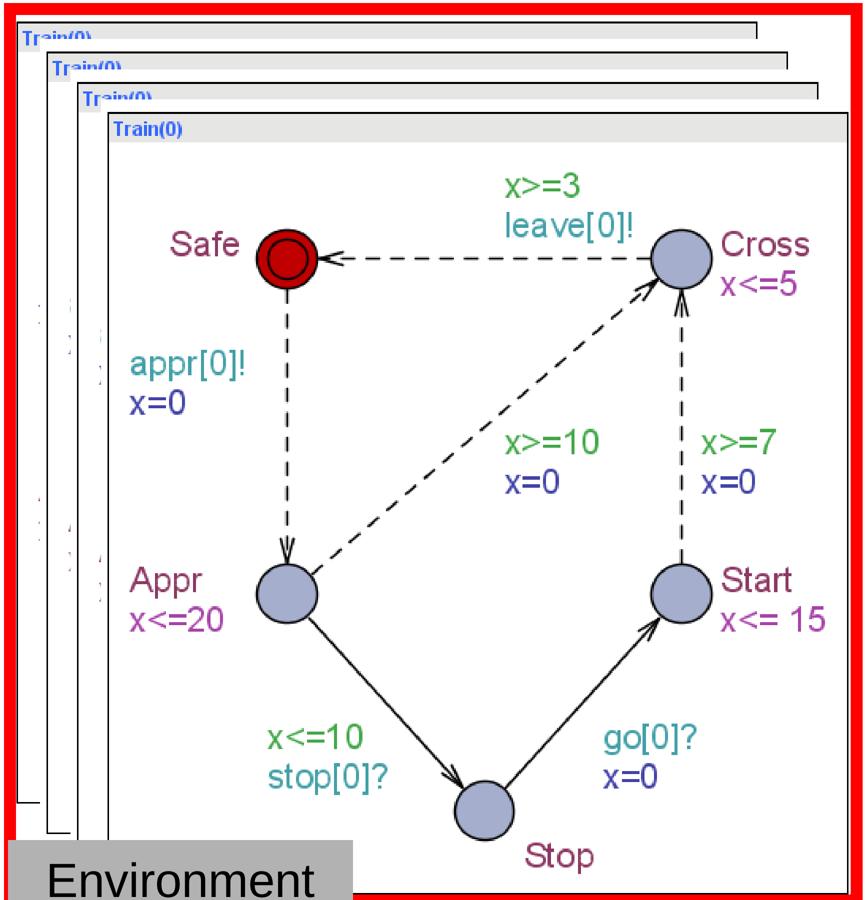

$\phi$: Never two trains at the crossing at the same time

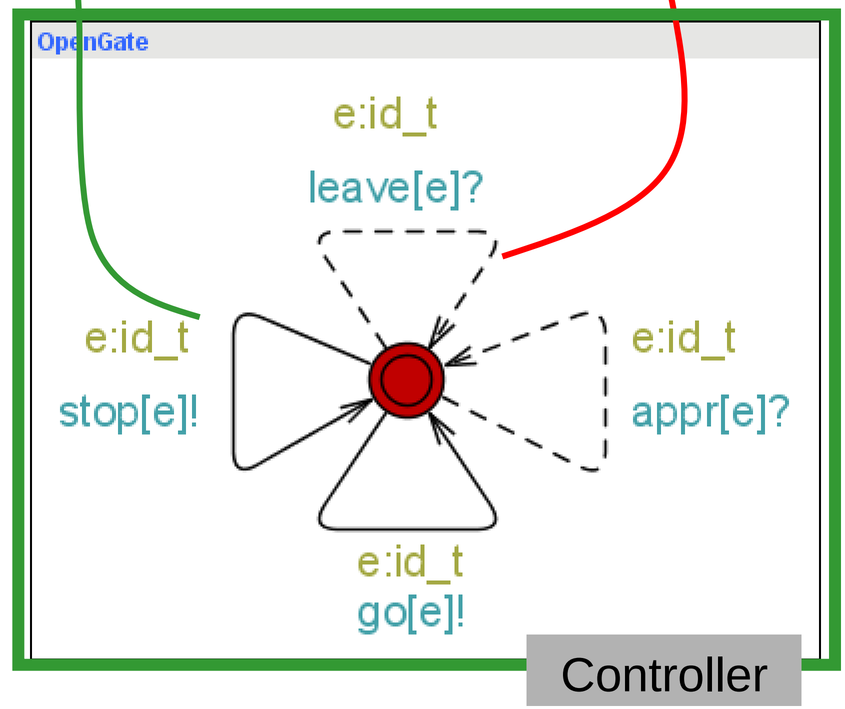
# Synthesis

Two Player Game

Controllable

Uncontrollable

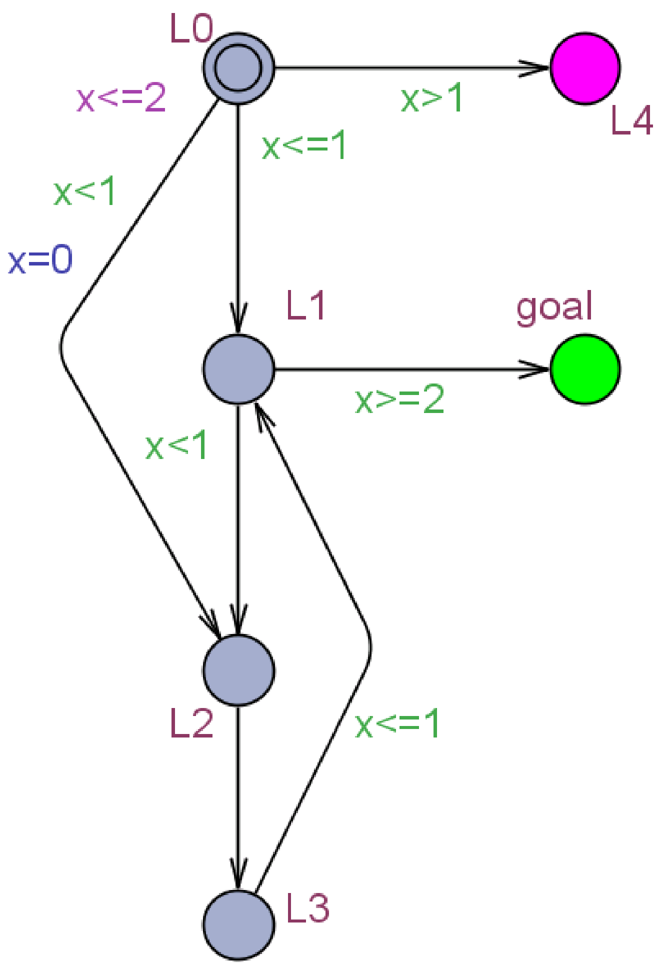



Find strategy for controllable actions st behaviour satisfies $\phi$

$\phi$: Never two trains at the crossing at the same time

# Timed Automata & Model Checking




**State** (L1, x=0.81)

**Transitions**

(L1 , x=0.81)

– 2.1 –>

(L1 , x=2.91)

–>

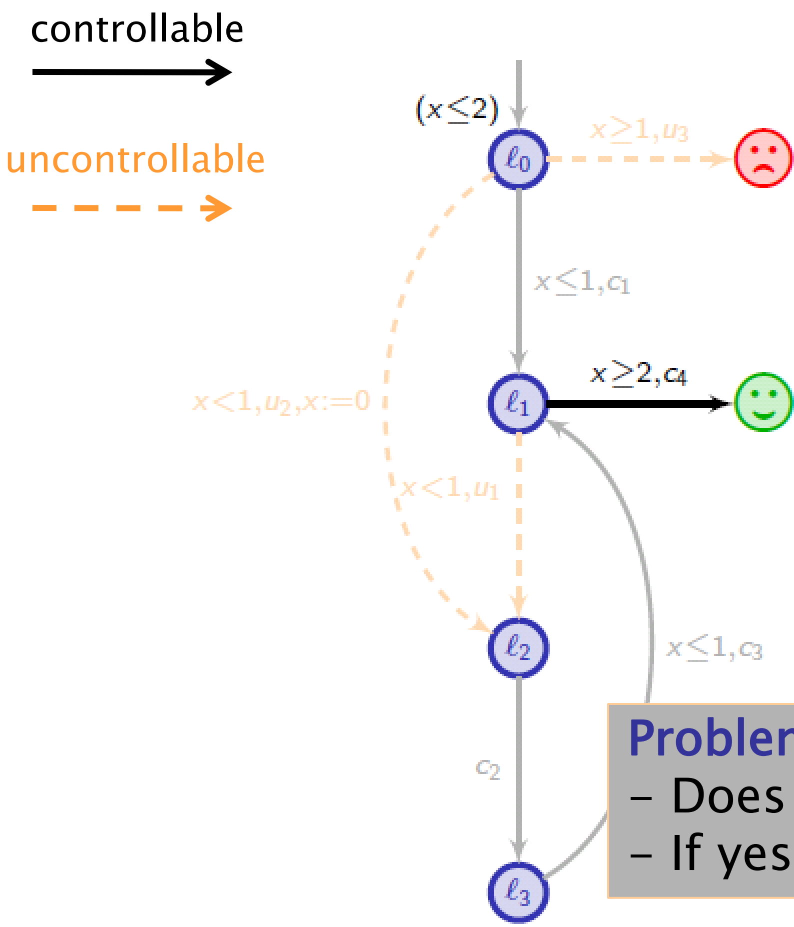
(goal , x=2.91)

E⟨⟩ goal ?

A⟨⟩ goal ?

A[ ] ¬ L4 ?

# Timed Game Automata & Synthesis




A (memoryless) winning strategy

- from $(\ell_0, 0)$, play $(0.5, c_1)$
  $\rightsquigarrow$ can be preempted by $u_2$

**Problems to be considered:**
- Does there exist a winning strategy?
- If yes, compute one (as simple as possible)

# Decidability of Timed Games

**Theorem [AMPS98,HK99]**

Reachability and safety timed games are decidable and EXPTIME-complete. Furthermore memoryless and "region-based" strategies are sufficient.

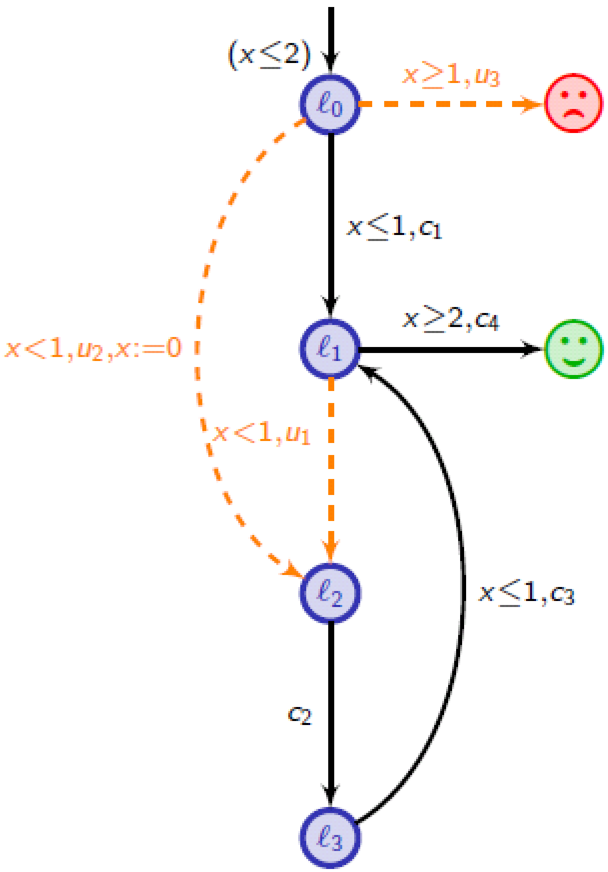
$\rightsquigarrow$ classical regions are sufficient for solving such problems

**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)*.
[BHPR07] Brihaye, Henzinger, Prabhu, Raskin. Minimum-time reachability in timed games *(ICALP'07)*.
[JT07] Jurdziński, Trivedi. Reachability-time games on timed automata *(ICALP'07)*.

# Computing Winning States



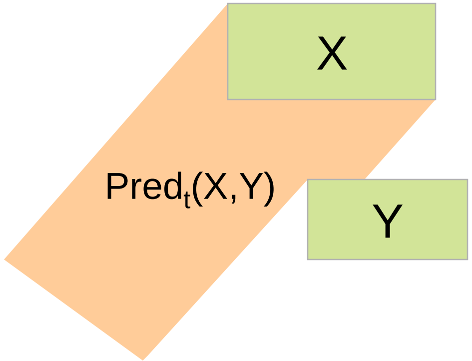

# Reachability Games

## Backwards Fixed-Point Computation

### Definitions

$$\text{cPred}(X) = \{\, q \in Q \mid \exists\, q' \in X.\ q \rightarrow_c q' \}$$
$$\text{uPred}(X) = \{\, q \in Q \mid \exists\, q' \in X.\ q \rightarrow_u q' \}$$
$$\text{Pred}_t(X,Y) = \{\, q \in Q \mid \exists\, t.\ q^t \in X \ \text{ and } \ \forall\, s \leq t.\ q^s \in Y^C \}$$

$$\pi(X) = \text{Pred}_t[\, X \cup \text{cPred}(X)\ ,\ \text{uPred}(X^C)\ ]$$




**Theorem:**
The set of winning states is obtained as the least fixpoint of the function: $X \mapsto \pi(X) \cup \text{Goal}$

# Symbolic On-the-fly Algorithms for Timed Games [CDF+05, BCD+07]

- $S, S', \ldots$
  are *symbolic* states, i.e. sets of concrete states;
- $G$
  is the set of (concrete) goal states;
- $E = \{S \xrightarrow{c} S', S \xrightarrow{u} S'\}$
  the (finite) set of symbolic transitions (controlla
- $Waiting \subseteq E$
  is the list of symbolic transitions waiting to be p
- $Passed$
  is the list of the passed symbolic states;
- $Win[S] \subseteq S$
  is the subset of $S$ currently known to be winnin
- $Depend[S] \subseteq E$
  indicates the edges (predecessors) of $S$ which mu
  information about $S$ is obtained.

**Initialization:**

$Passed \leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \vec{0})\}^{\nearrow}$;
$Waiting \leftarrow \{(S_0, \alpha, S') \mid S' = \mathsf{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\mathsf{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

**Main:**

**while** $((Waiting \neq \emptyset) \wedge (s_0 \notin Win[S_0]))$ **do**
  $e = (S, \alpha, S') \leftarrow pop(Waiting)$;
  **if** $S' \notin Passed$ **then**
    $Passed \leftarrow Passed \cup \{S'\}$;
    $Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
    $Win[S'] \leftarrow S' \cap (\{\mathsf{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
    $Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \mathsf{Post}_\alpha(S')^{\nearrow}\}$;
    **if** $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  **e** (* **reevaluate** *)$^a$
    $Win^* \leftarrow \mathsf{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \mathsf{Pred}_c(Win[T]),$
      $\bigcup_{S \xrightarrow{u} T} \mathsf{Pred}_u(T \setminus Win[T])) \cap S$;
    **if** $(Win[S] \subsetneq Win^*)$ **then**
      $Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
    $Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
  **endif**
**endwhile**

symbolic version of on-the-fly MC algorithm
for modal mu-calculus
Liu & Smolka 98

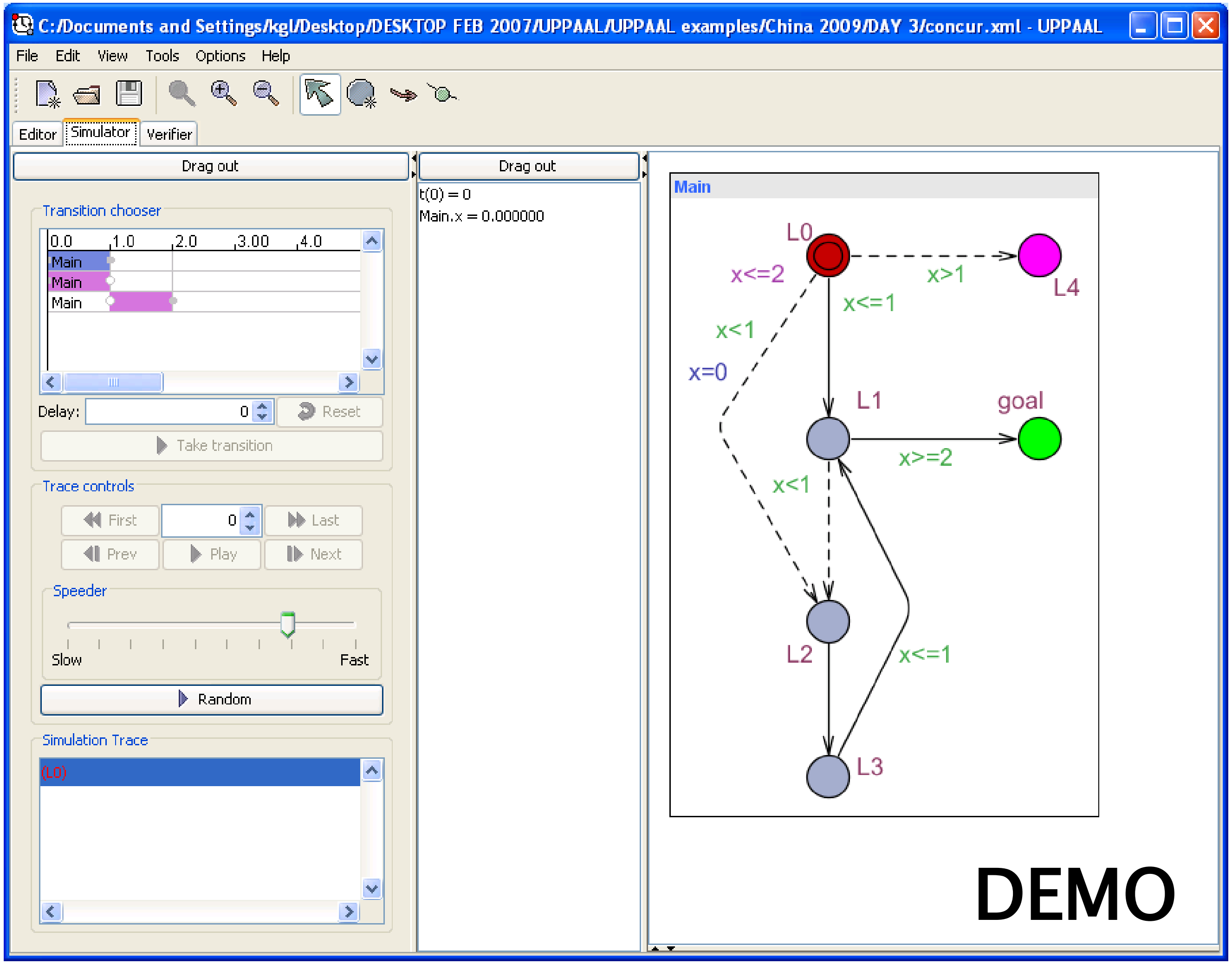

[CDF+05] Cassez, David, Fleury, Larsen, Lime. Efficient on-the-fly algorithms for the analysis of timed games (CONCUR'05).
[BCD+07] Berhmann, Cougnard, David, Fleury, Larsen, Lime. Uppaal-Tiga: Time for playing games! (CAV'07).

# UPPAAL Tiga [CDF+05, BCD+07]

- Reachability properties:
  - control: A[ p U q ] *until*
  - control: A⟨⟩ q ⇔ control: A[ true U q ]
- Safety properties:
  - control: A[ p W q ] *weak until*
  - control: A[] p ⇔ control: A[ p W false ]
- Time-optimality :
  - control_t*(u,g): A[ p U q ]
    - u is an upper-bound to prune the search
    - g is the time to the goal from the current state

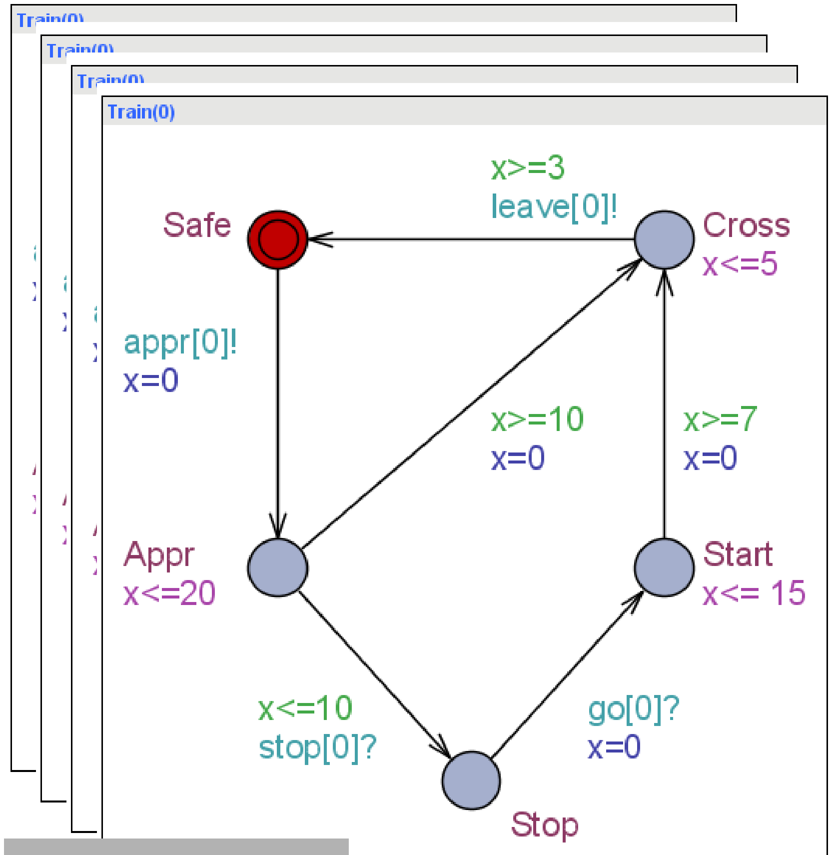
[CDF+05] Cassez, David, Fleury, Larsen, Lime. Efficient on-the-fly algorithms for the analysis of timed games *(CONCUR'05)*.
[BCD+07] Berhmann, Cougnard, David, Fleury, Larsen, Lime. Uppaal-Tiga: Time for playing games! *(CAV'07)*.

# UPPAAL Tiga

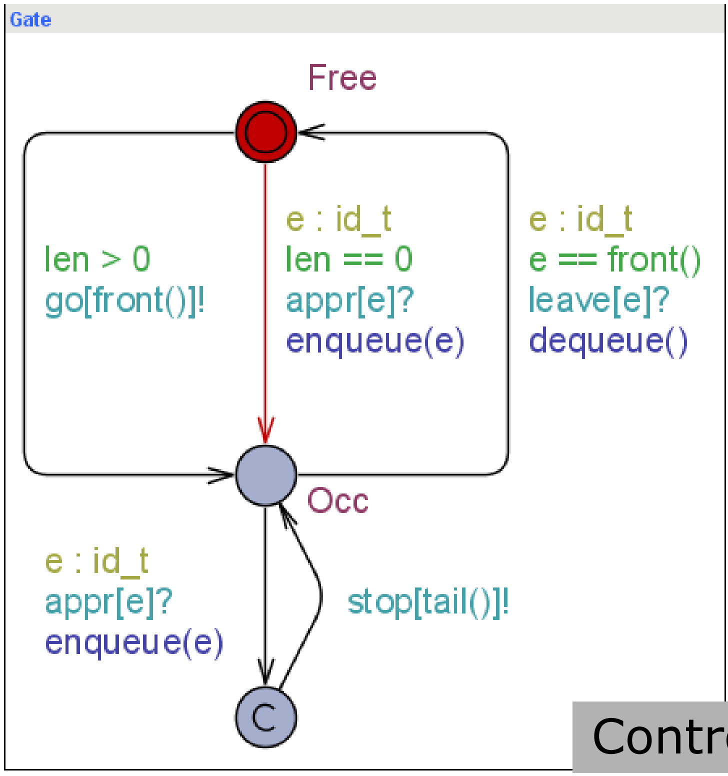



DEMO

# Model Checking (ex Train Gate)



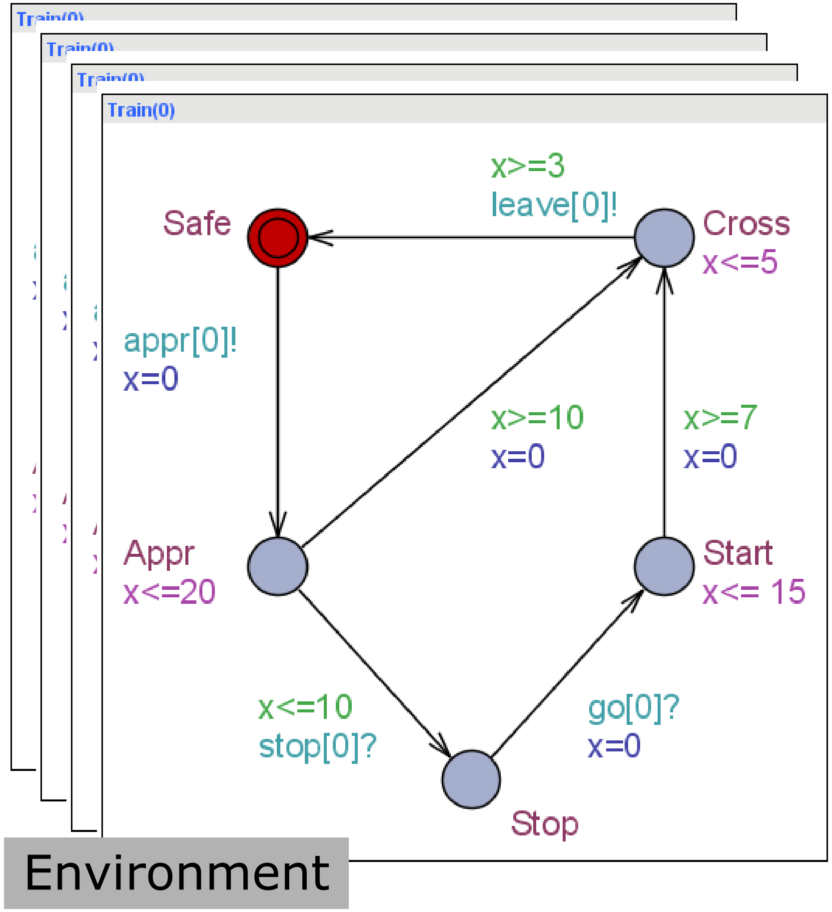

Environment

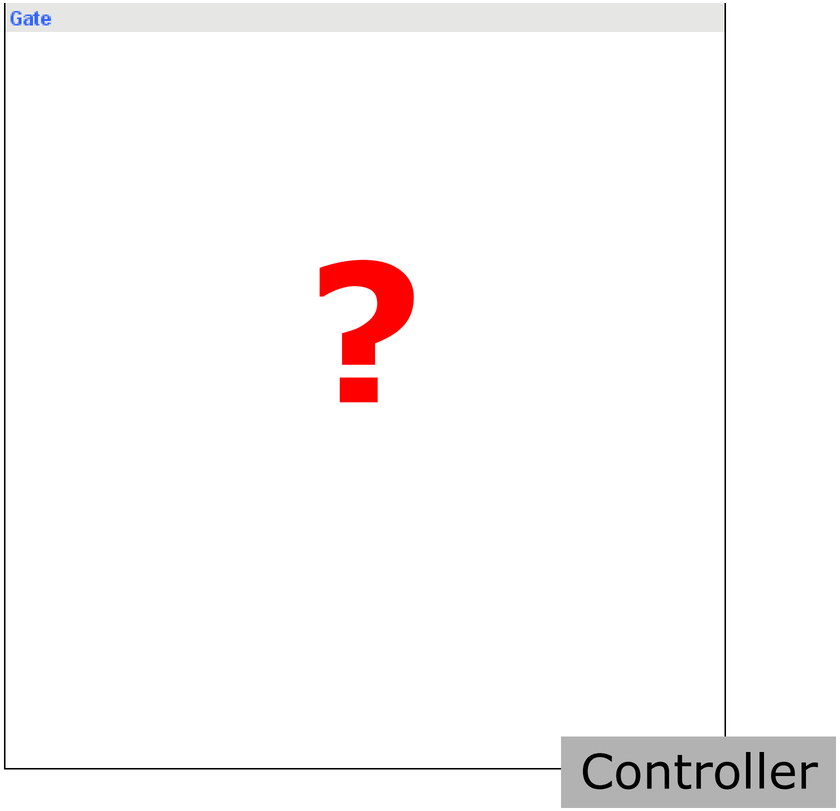



Controller

φ: Never two trains at the crossing at the same time

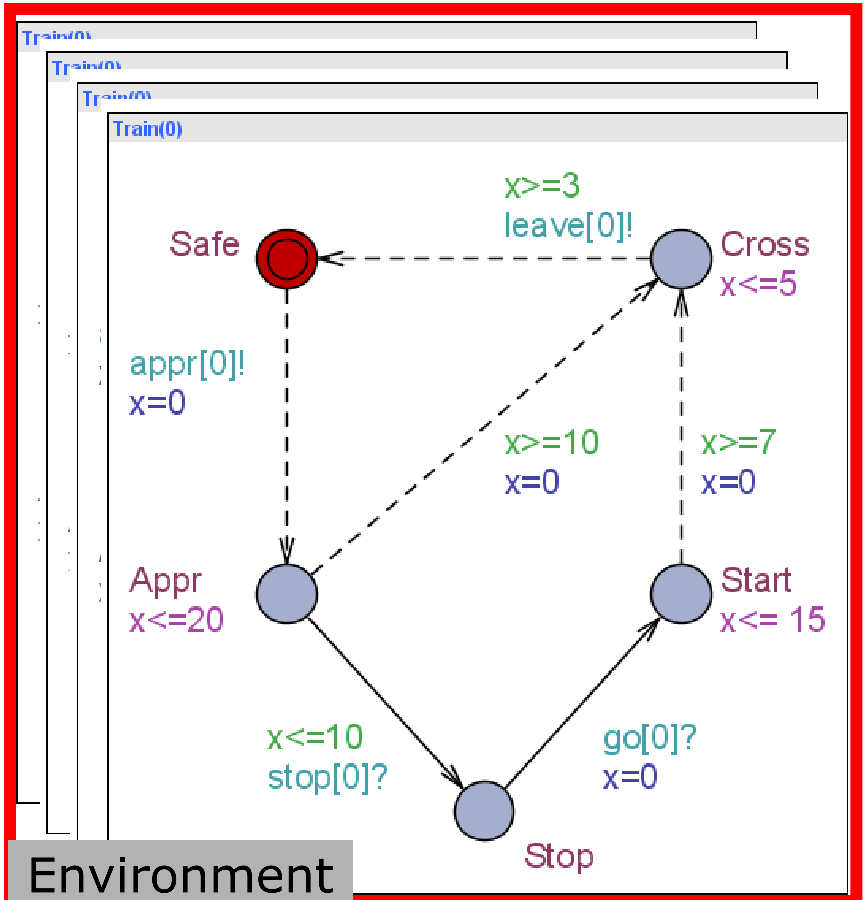
# Synthesis (ex Train Gate)

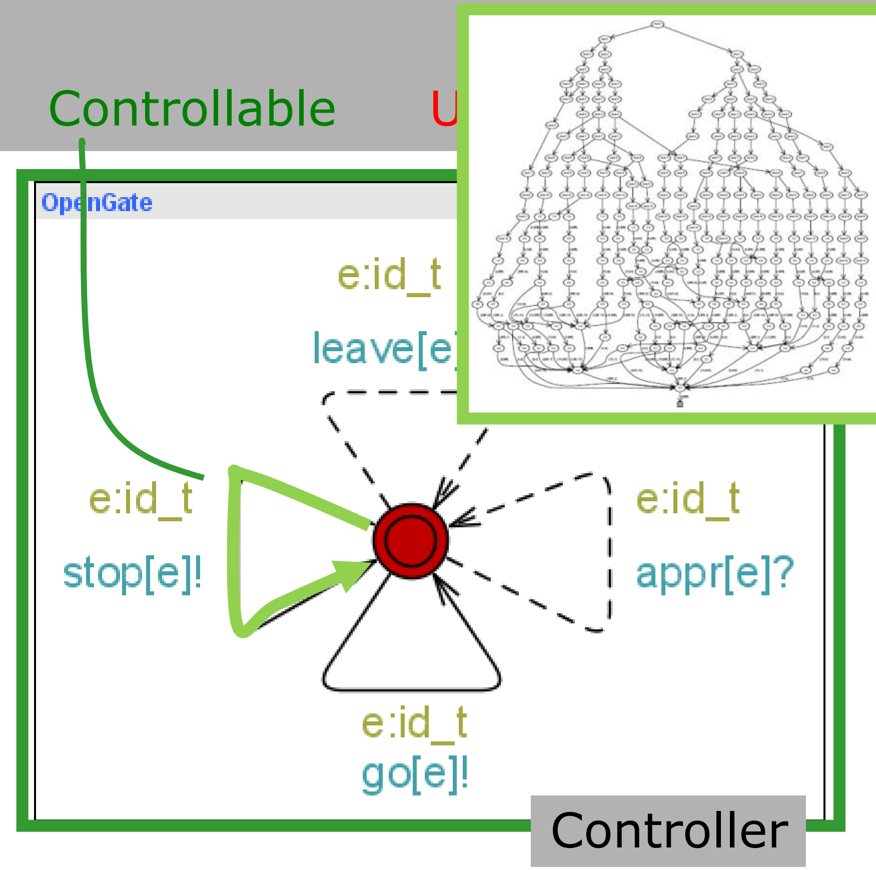



$\phi$: Never two trains at the crossing at the same time

# Timed Games



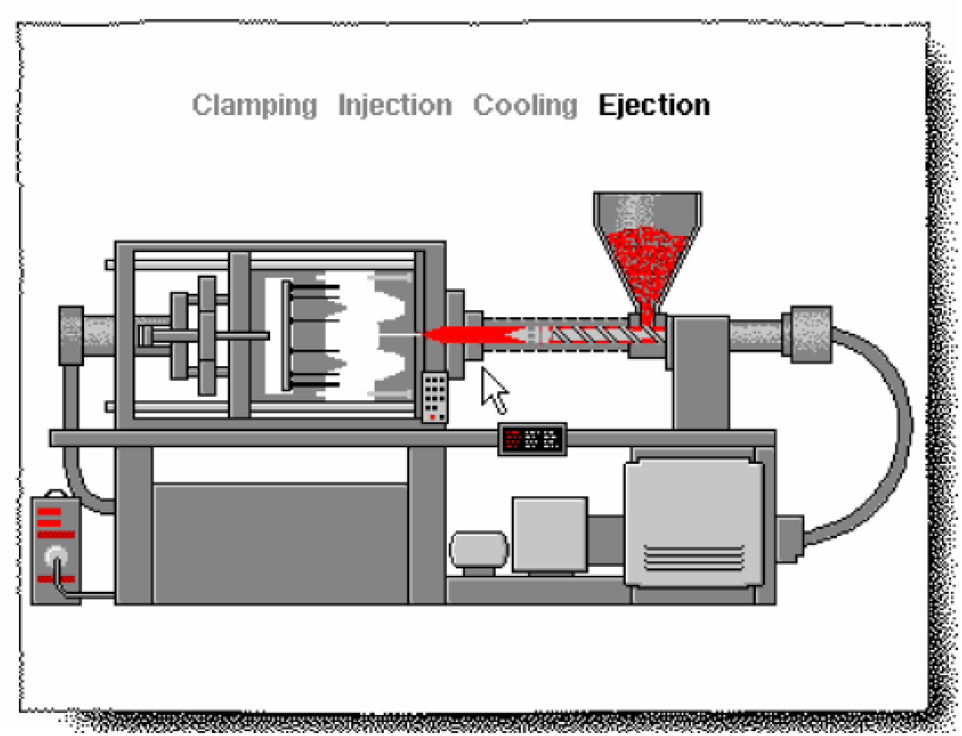

Find strategy for controllable actions st behaviour satisfies $\phi$

$\phi$: Never two trains at the crossing at the same time

# Plastic Injection Molding Machine

Quasiomodo

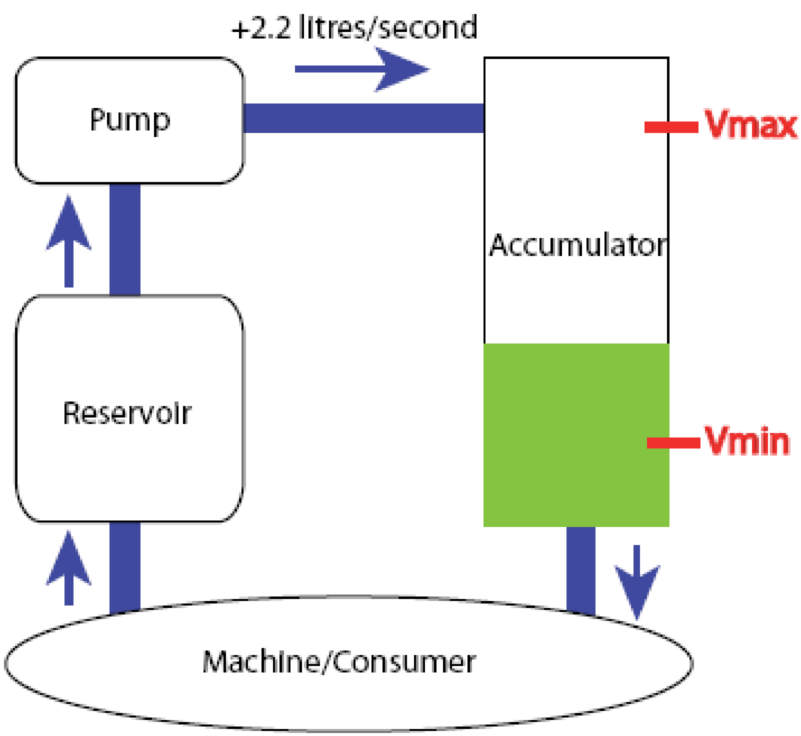
[CJL+09]




- Robust and optimal control
- Tool Chain
  - Synthesis: UPPAAL TIGA
  - Verification: PHAVer
  - Performance: SIMULINK
- 40% improvement of existing solutions..

[CJL+09] Cassez, Jessen, Larsen, Raskin, Reynier.Automatic Synthesis of Robust and Optimal Controllers – An Industrial Case Study *(HSCC'09)*.

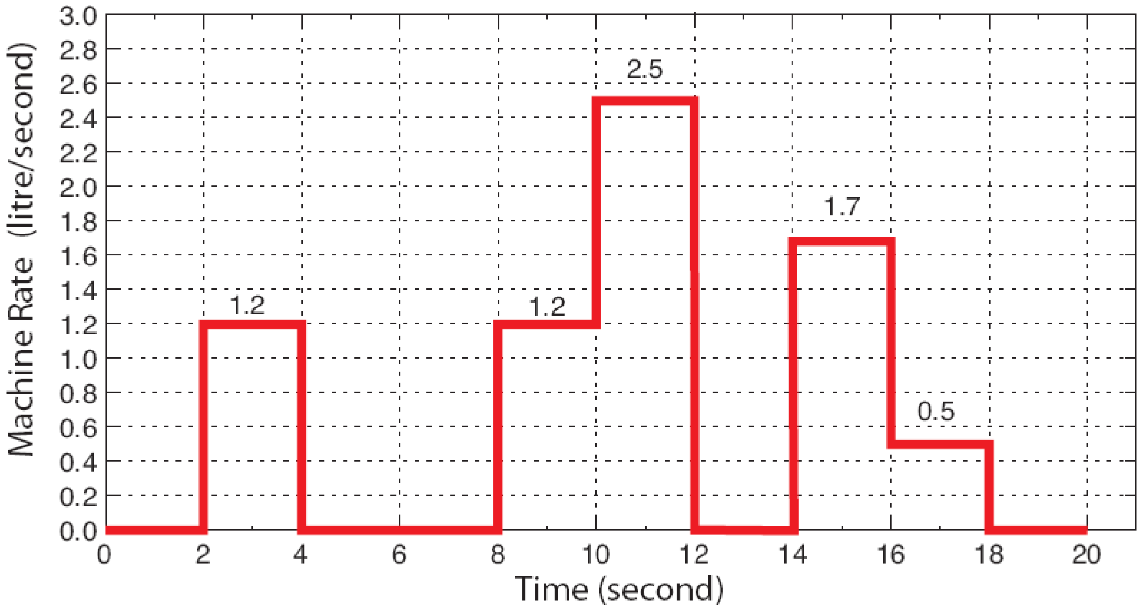
# Oil Pump Control Problem



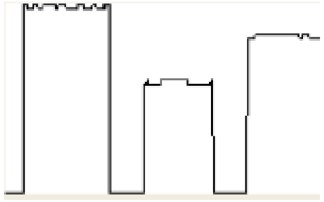

- R1: stay within safe interval [4.9,25.1]
- R2: minimize average/overall oil volume

$$\int_{t=0}^{t=T} v(t)dt \;/\; T$$

# The Machine (consumption)




- Infinite cyclic demand to be satisfied by our control strategy.
- P: latency 2 s between state change of pump
- F: noise 0.1 l/s



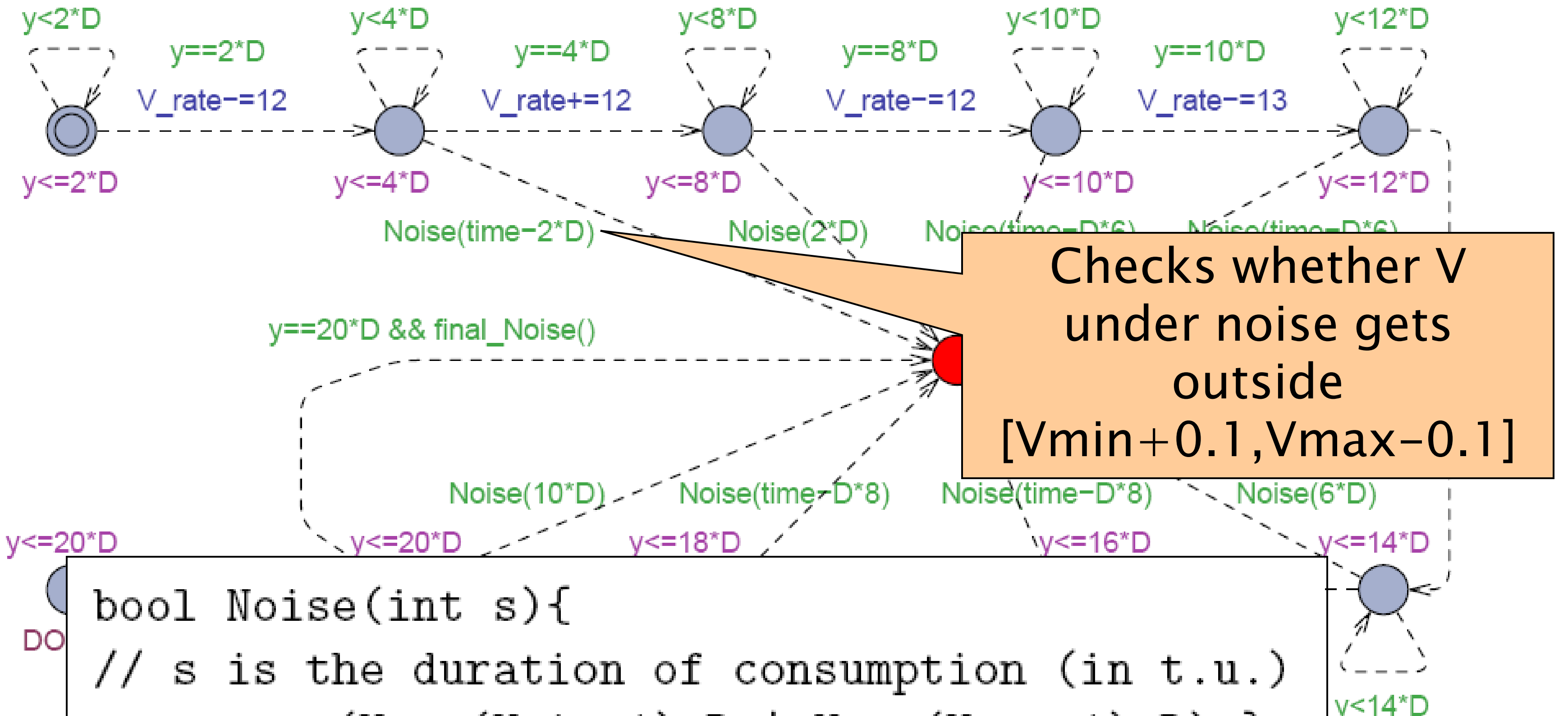
# Abstract Game Model

- UPPAAL Tiga
  offers games of perfect information

- Abstract game model such that states only contain information about:
  - Volume of oil at the beginning of cycle
  - The ideal volume as predicted by the consumption cycle
  - Current time within the cycle
  - State of the Pump (on/off)
  - Discrete model

D
V, V_rate
V_acc
time

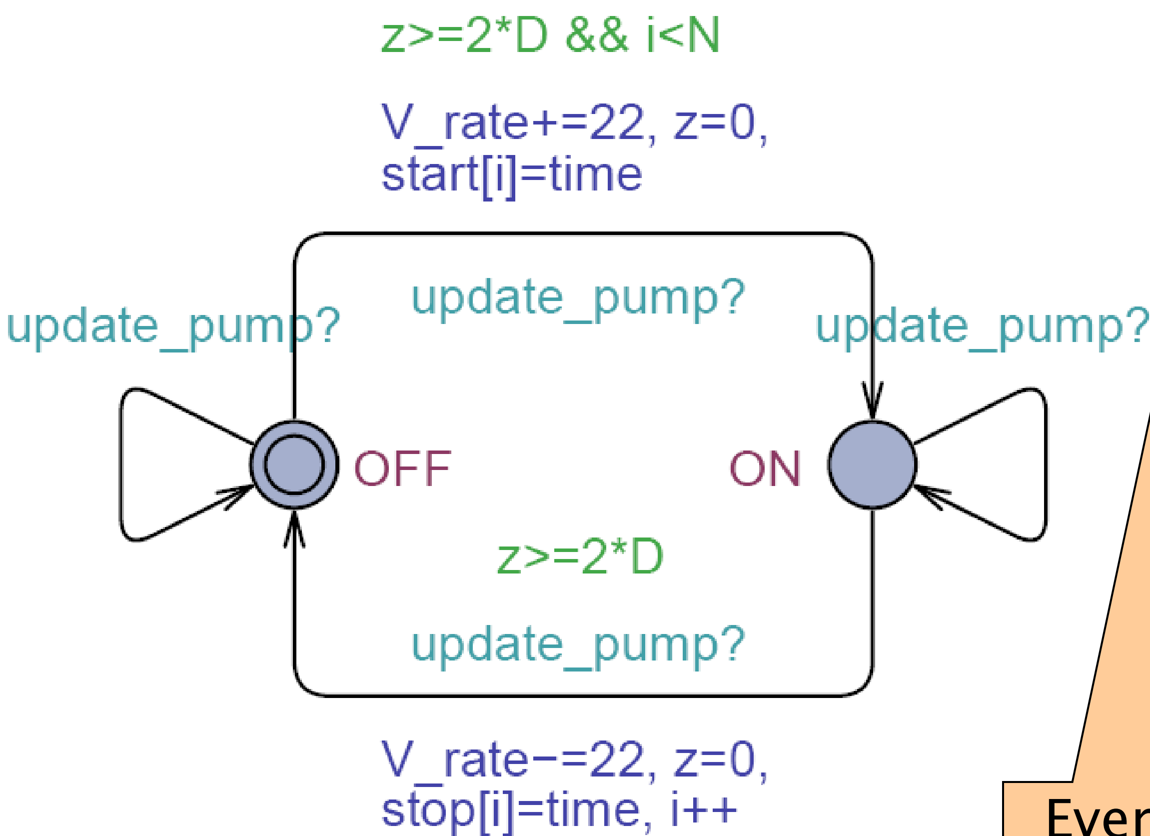
# Machine (uncontrollable)




Checks whether V under noise gets outside [Vmin+0.1,Vmax-0.1]

```
bool Noise(int s){
// s is the duration of consumption (in t.u.)
  return (V-s<(Vmin+1)*D | V+s>(Vmax-1)*D);}
```

# **Pump** (controllable)

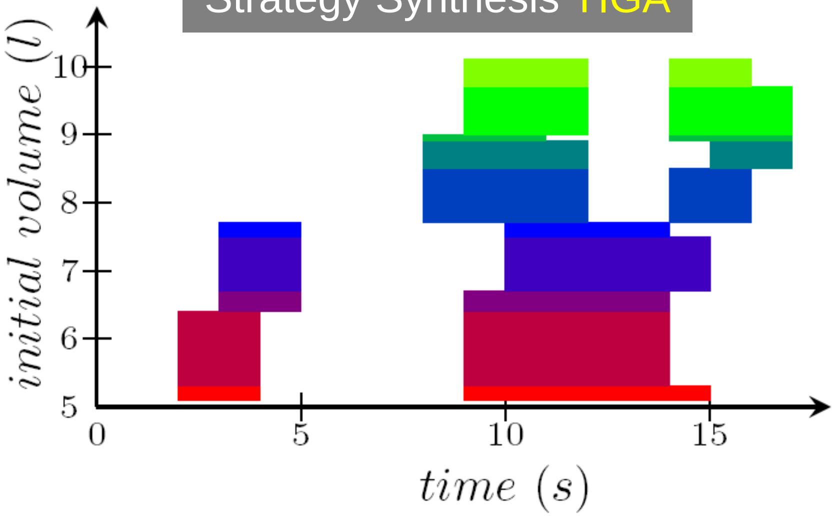


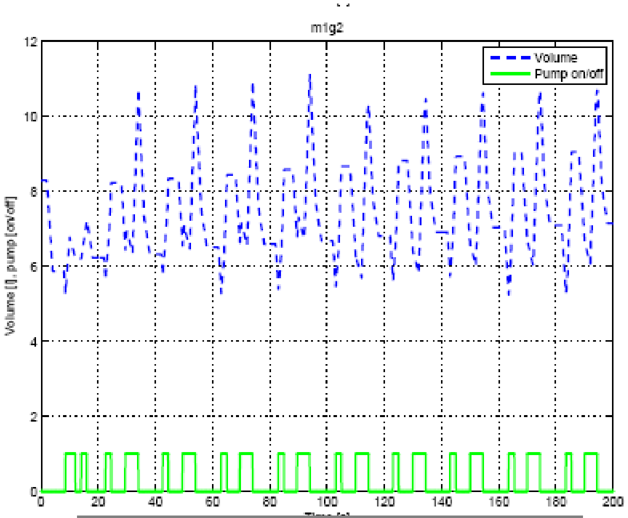

```
void update_val(){
   int V_pred = V;
   time++;
   V+=V_rate;
   V_acc+=V+V_pred;
}
```

Every 1 (one) seconds

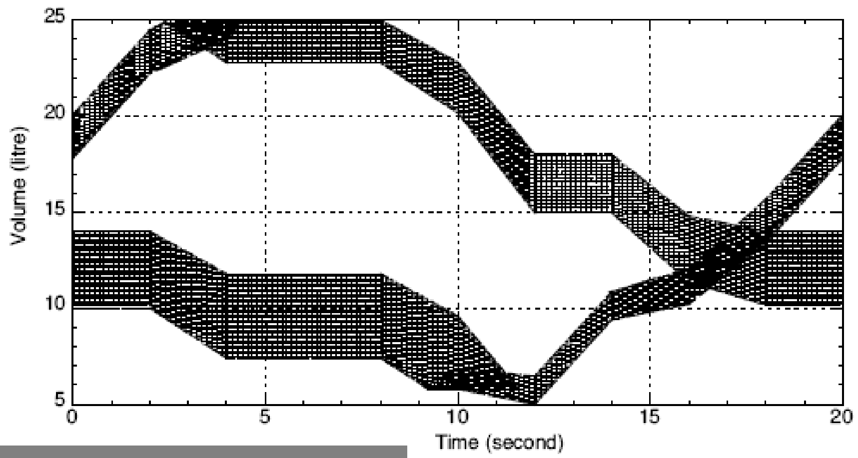
# Tool Chain

Strategy Synthesis TIGA




Performance Evaluation SIMULINK




Verification PHAVER

**Guaranteed**

Correctness

Robustness

with

40% Improvement

# LAB Exercises

www.cs.aau.dk/~kgl/Shanghai2013

Exercise 28 (Jobshop Scheduling Part 1)
Exercise 19 (Train Gate Part 1)

AALBORG UNIVERSITY DENMARK