AMETIST
advanced methods for timed systems

Optimal & Real Time Scheduling

using

Model Checking Technology

Kim G. Larsen

AALBORG UNIVERSITY DENMARK

BRICS
Basic Research
in Computer Science

CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

AIPS – a partial observation

BRICS
Basic Research
in Computer Science

CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

AMETIST
advanced methods for timed systems

April 2002 – June 2005 IST-2001-35304

- **Academic partners:**
  - Nijmegen
  - Aalborg
  - Dortmund
  - Grenoble
  - Marseille
  - Twente
  - Weizmann
- **Industrial Partners**
  - Axxom
  - Bosch
  - Cybernetix
  - Terma

BRICS
Basic Research
in Computer Science

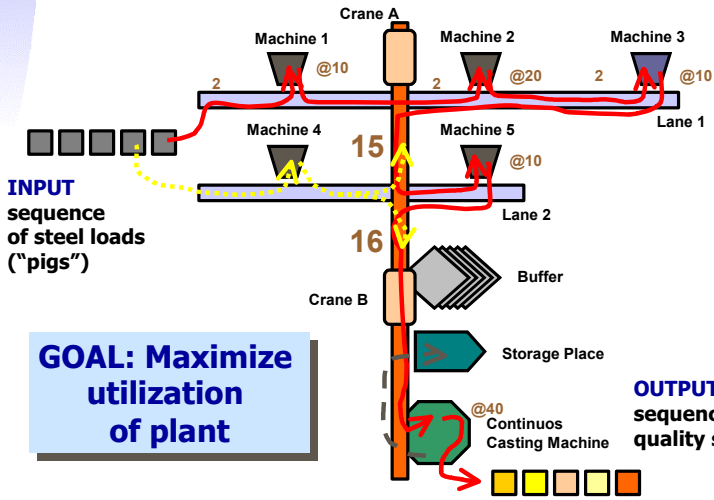
AMETIST
advanced methods for timed systems

OBJECTIVES

- powerful, unifying mathematical modelling
- efficient computerized problem-solving tools
- distributed real-time systems
- time-dependent behaviour and dynamic resource allocation
- **TIMED AUTOMATA**

BRICS
Basic Research
in Computer Science

SIDMAR Overview



INPUT
sequence
of steel loads
("pigs")

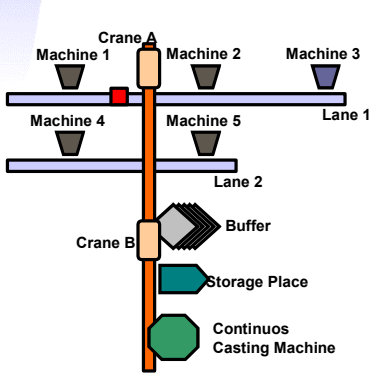
GOAL: Maximize utilization of plant

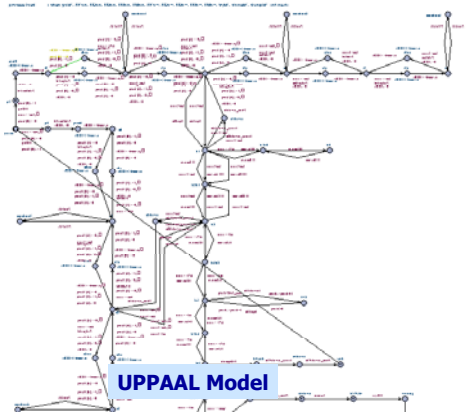
OUTPUT
sequence of higher quality steel

LTR Project VHS (Verification of Hybrid systems)

SIDMAR Modelling

A Single Load





UPPAAL Model

AMETIST
advanced methods

Case S

- Cyberr
  - Sma
    Pers
- Terma:
  - Mem
- Bosch:
  - Car
    Sens
- AXXOM
  - Lacq
- Benchmarks

UPPAAL
UPPSALA UNIVERSITET AALBORG UNIVERSITY
Copyright 1995-2003 by Uppsala University and Aalborg University. All rights reserved.
More information at http://www.uppaal.com.
UPPAAL 3.4.7, Aug 2004.

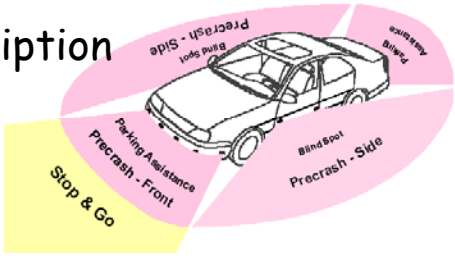
CLASSIC

UPPAAL
UPPSALA UNIVERSITET AALBORG UNIVERSITY
Copyright 1995-2003 by Uppsala University and Aalborg University. All rights reserved.
More information at http://www.uppaal.com.
UPPAAL CORA 041109, Nov 2004.

CORA

BRICS
Basic Research
in Computer Science

CPS: Informal description

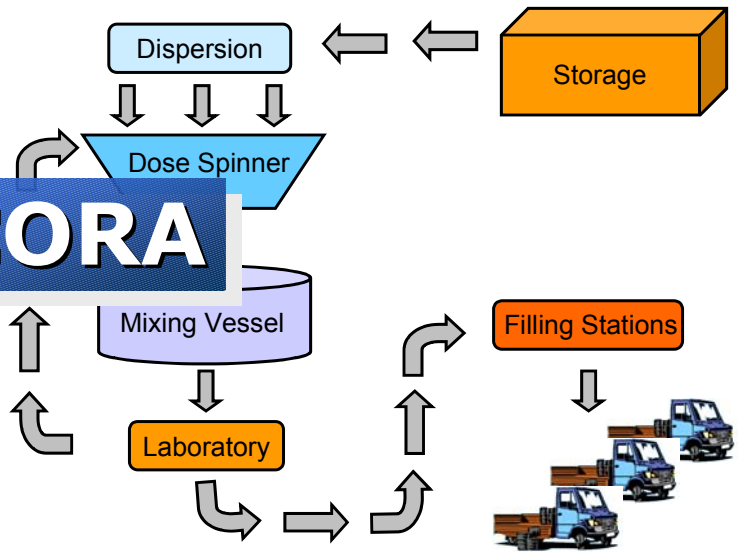


- CPS obtains and makes available for other systems information about environment of a car. This information may be used for:
  - Parking assistance
  - Pre-crash detection
  - Blind spot supervision
  - Lane change assistance
  - Stop & go
  - Etc
- The CPS considered in this case study
  - One sensor group only (currently 2 sensors)
  - Only the front sensors and corresponding controllers
  - Application: pre-crash detection, parking assistance, stop & go

Eg Brinksma — Car Periphery Supervision System: Case Study 3 — 2

Product flow of a Product

axxom



05.06.2005 — Axxom Software AG — Seite: 3

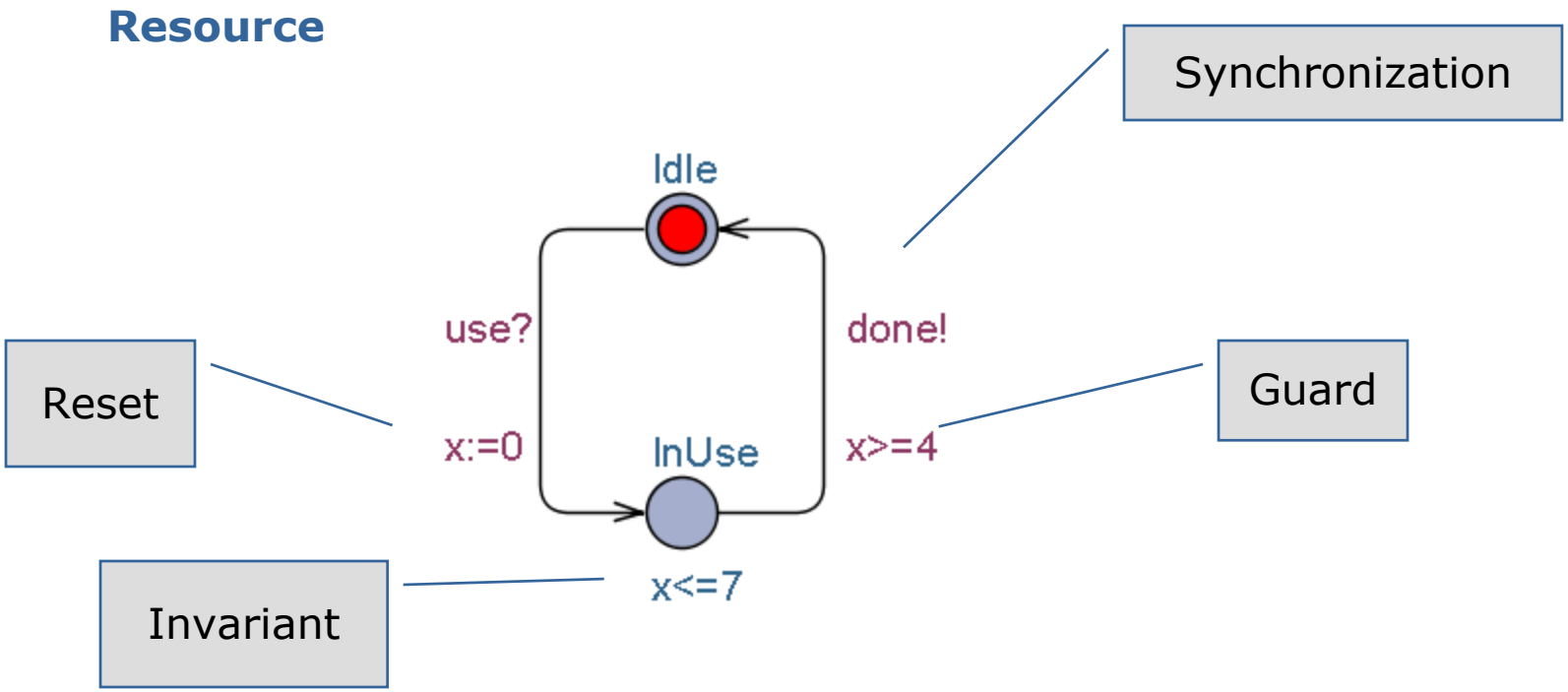
Overview

- Timed Automata & Scheduling
- Priced Timed Automata and Optimal Scheduling
- AMETIST Results
- Beyond Static Scheduling

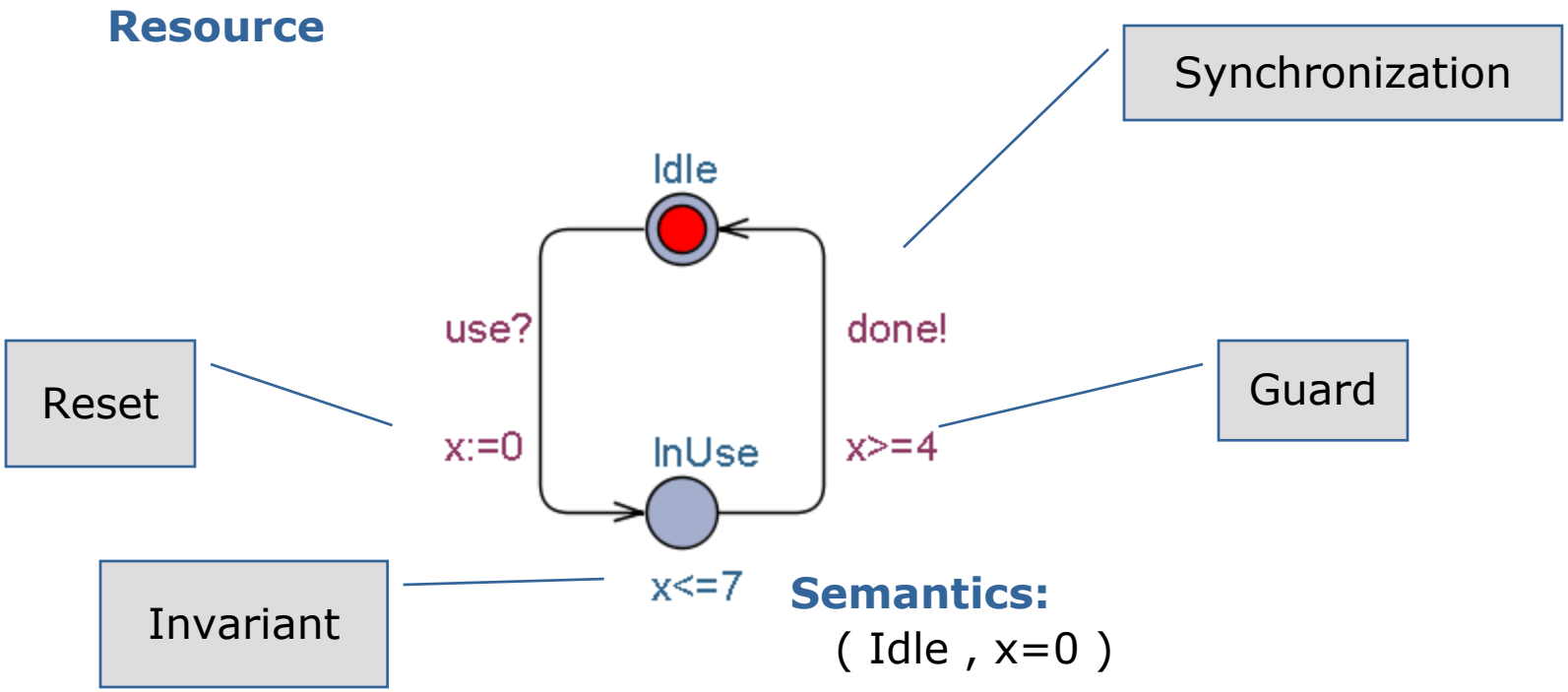
Overview

- Timed Automata & Scheduling
- Priced Timed Automata and Optimal Scheduling
- AMETIST Results
- Beyond Static Scheduling

Timed Automata [Alur & Dill'89]

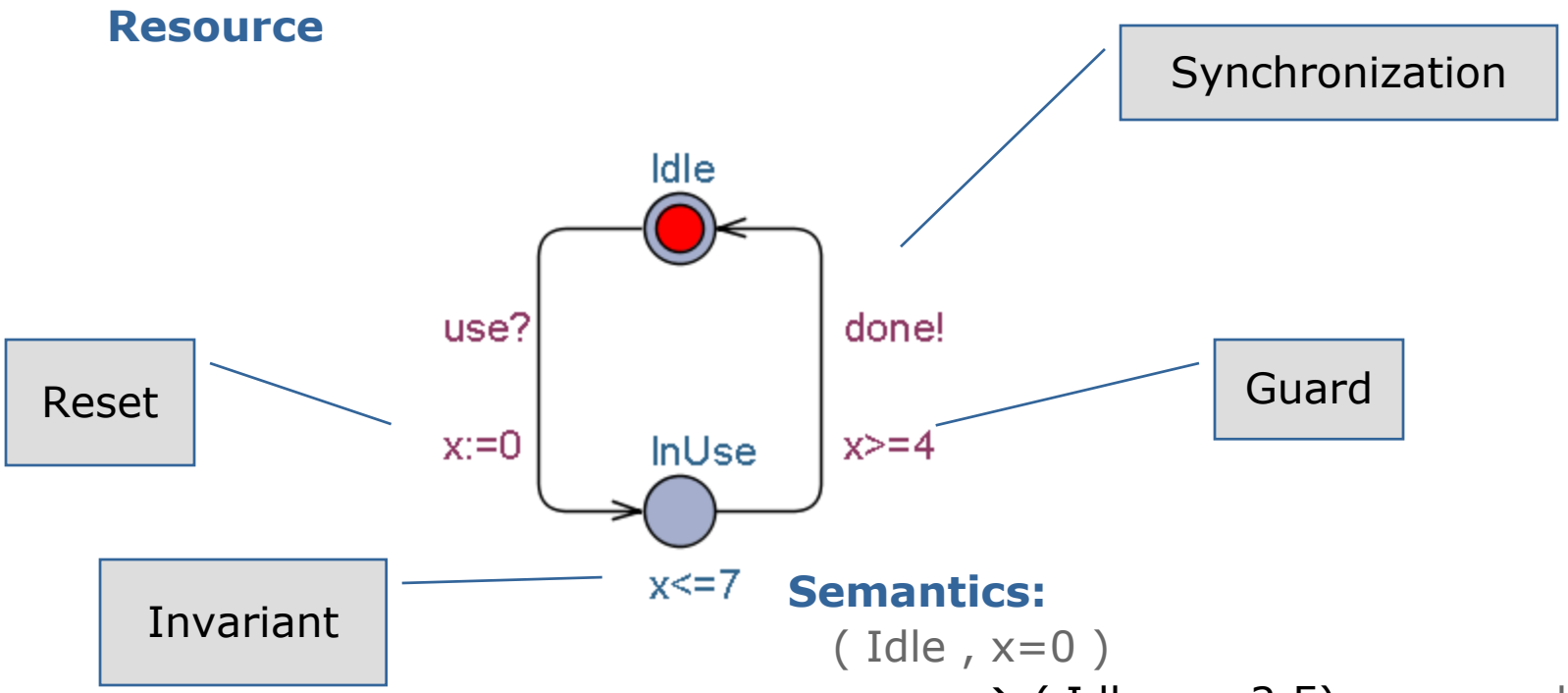


Timed Automata [Alur & Dill'89]



BRICS
Basic Research
in Computer Science

Timed Automata [Alur & Dill'89]



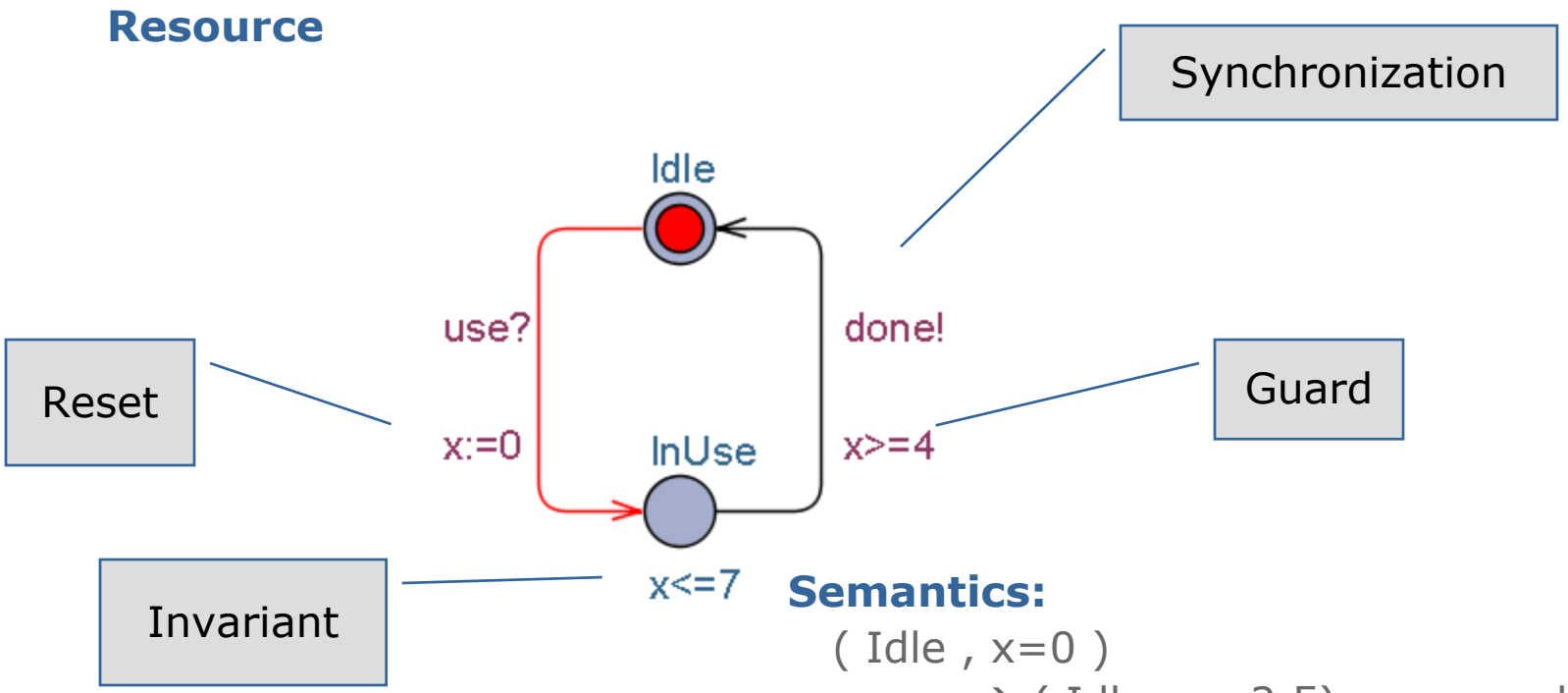
Semantics:

(Idle , x=0)

→ (Idle , x=2.5) d(2.5)

BRICS Basic Research in Computer Science

Timed Automata [Alur & Dill'89]



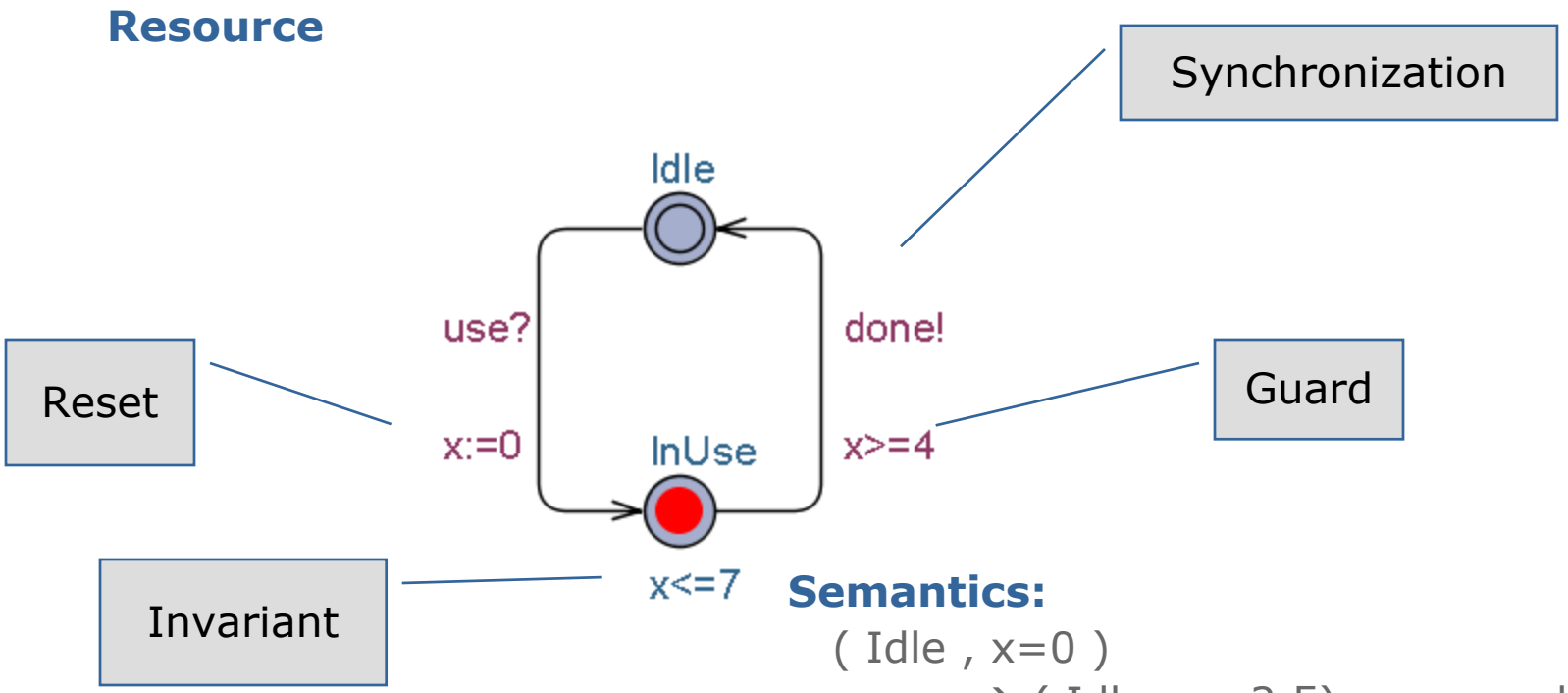
Semantics:

(Idle , x=0)

→ (Idle , x=2.5) d(2.5)

→ (InUse , x=0) use?

Timed Automata [Alur & Dill'89]



Semantics:

(Idle , x=0)

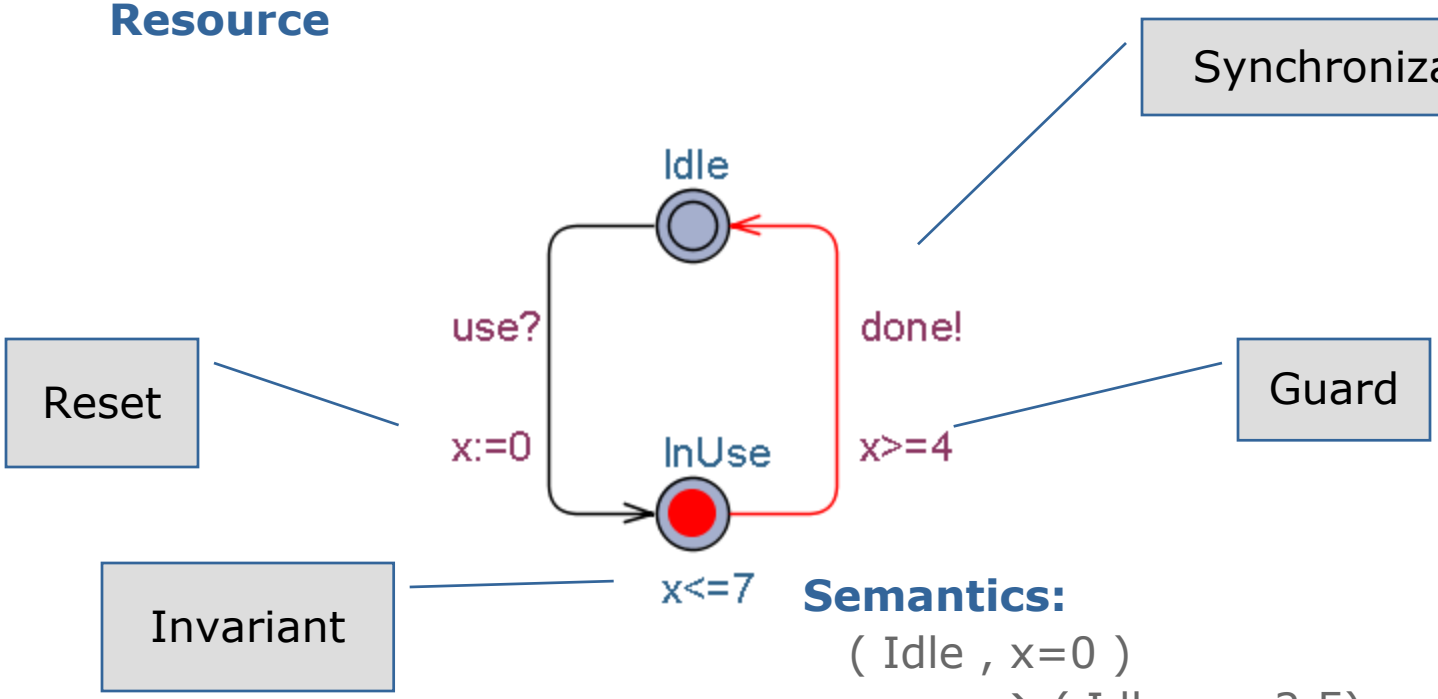
→ (Idle , x=2.5) d(2.5)

→ (InUse , x=0) use?

→ (InUse , x=5) d(5)

Timed Automata [Alur & Dill'89]

Resource

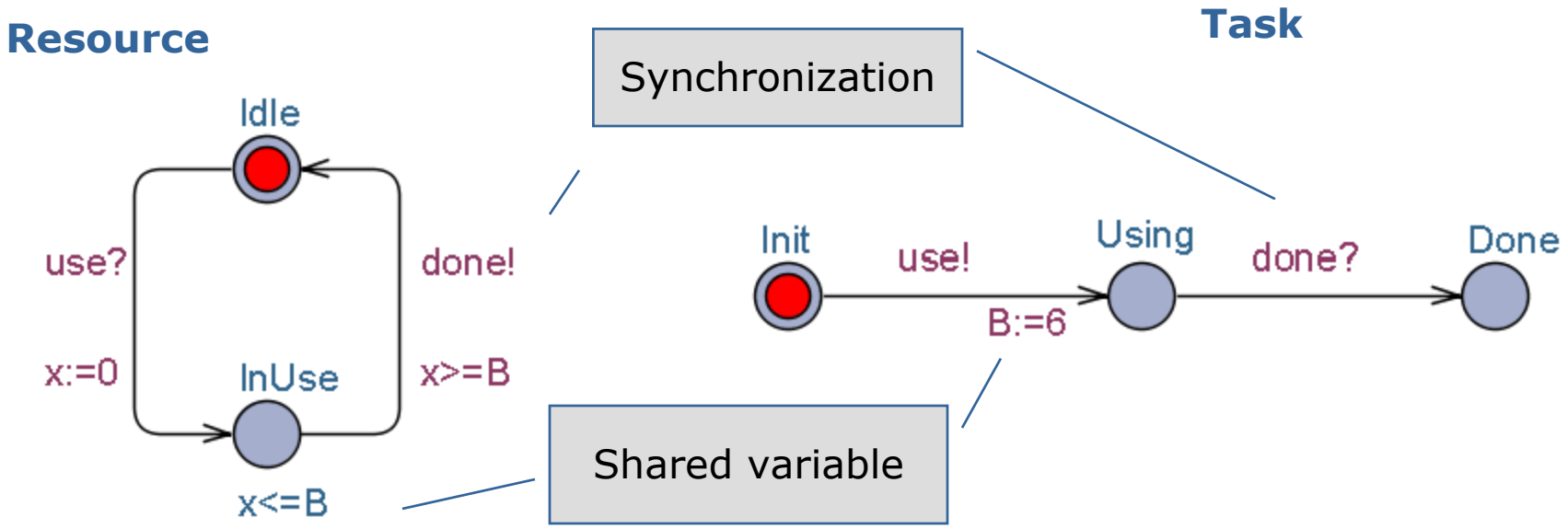


Semantics:

(Idle , x=0)

→ (Idle , x=2.5)	d(2.5)
→ (InUse , x=0)	use?
→ (InUse , x=5)	d(5)
→ (Idle , x=5)	done!
→ (Idle , x=8)	d(3)
→ (InUse , x=0)	use?

Composition



Semantics:

(Idle , Init , B=0, x=0)

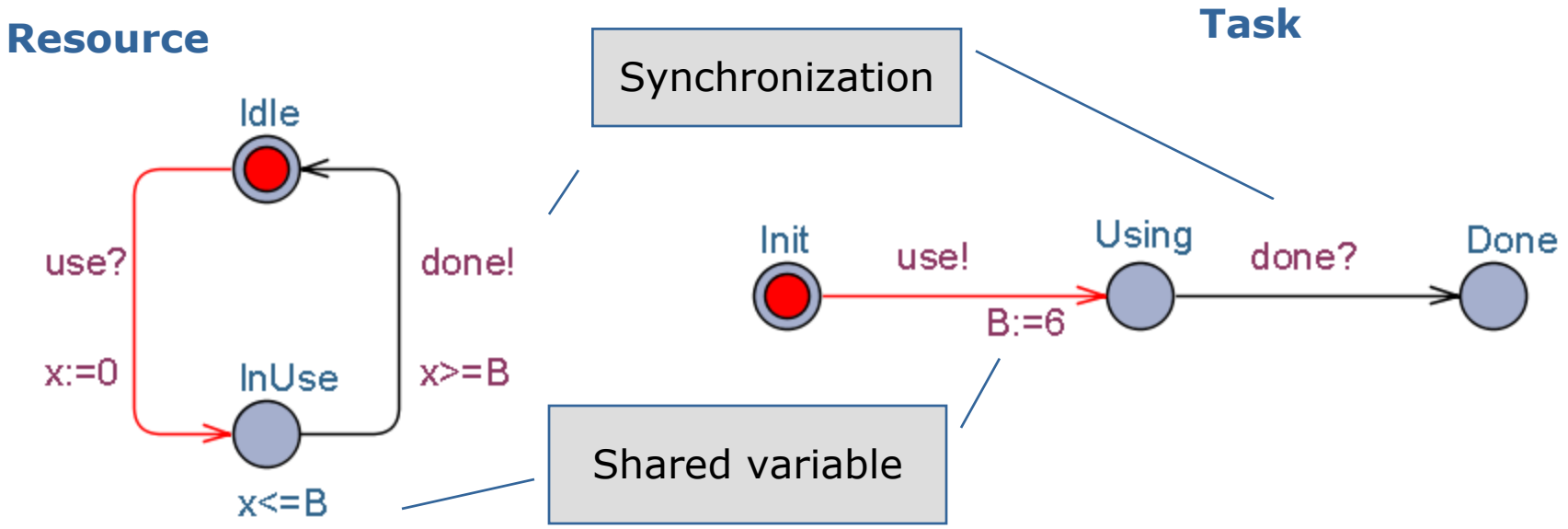
→ (Idle , Init , B=0 , x=3.1415) d(3)

→ (InUse , Using , B=6, x=0) use

→ (InUse , Using , B=6, x=6) d(6)

→ (Idle , Done , B=6 , x=6) done

Composition



Semantics:

(Idle , Init , B=0, x=0)

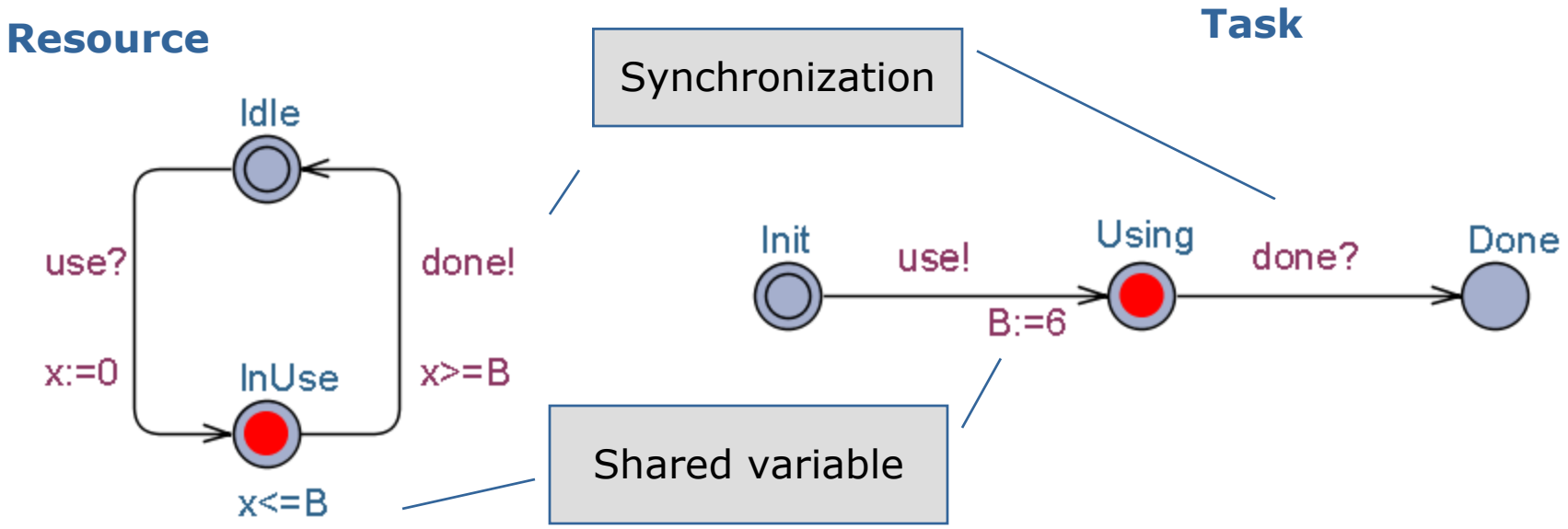
→ (Idle , Init , B=0 , x=3.1415) d(3)

→ (InUse , Using , B=6, x=0) use

→ (InUse , Using , B=6, x=6) d(6)

→ (Idle , Done , B=6 , x=6) done

Composition



Semantics:

(Idle , Init , B=0, x=0)

→ (Idle , Init , B=0 , x=3.1415) d(3)

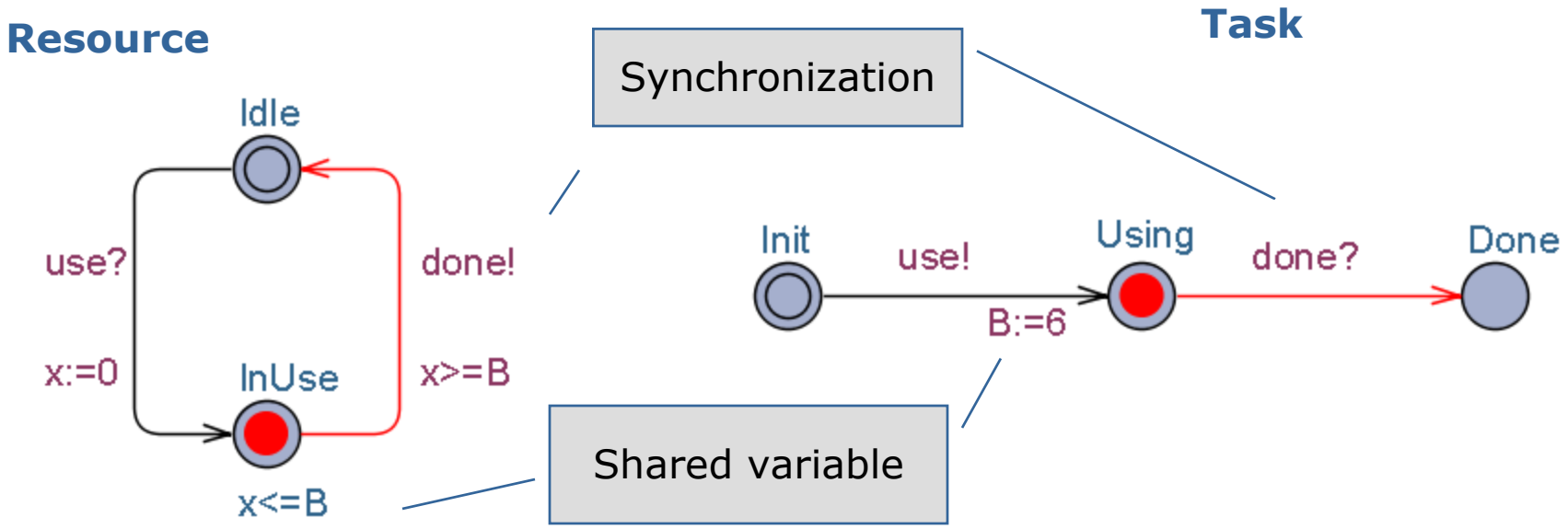
→ (InUse , Using , B=6, x=0) use

→ (InUse , Using , B=6, x=6) d(6)

→ (Idle , Done , B=6 , x=6) done

BRICS Basic Research in Computer Science

Composition



Semantics:

(Idle , Init , B=0, x=0)

→ (Idle , Init , B=0 , x=3.1415) d(3)

→ (InUse , Using , B=6, x=0) use

→ (InUse , Using , B=6, x=6) d(6)

→ (Idle , Done , B=6 , x=6) done

Jobshop Scheduling

RESOURCES

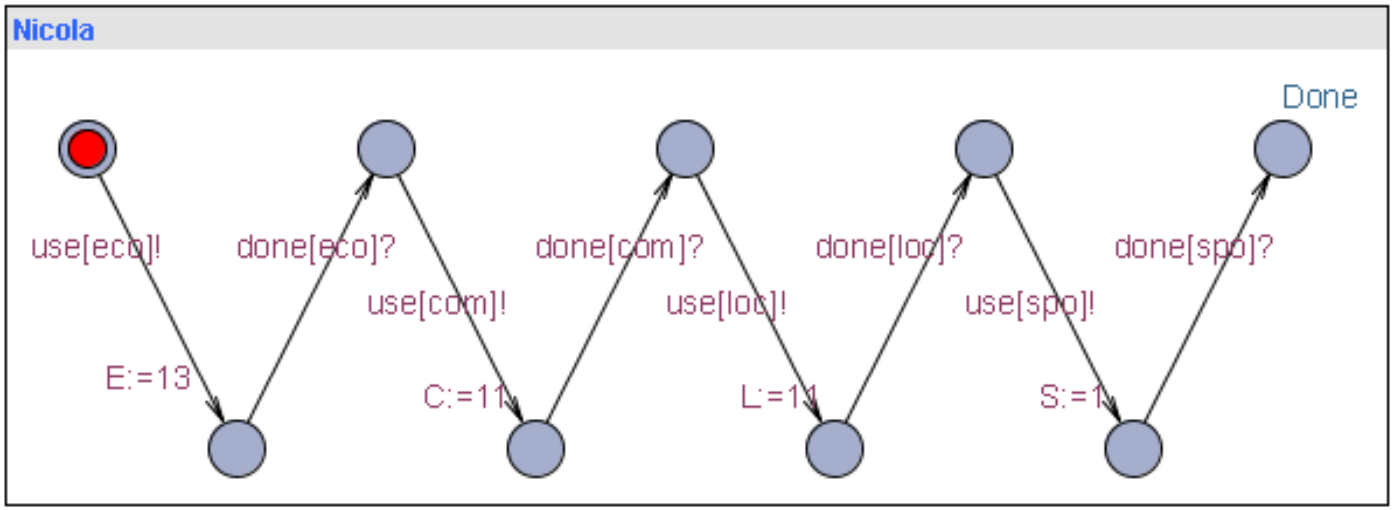
JOBs

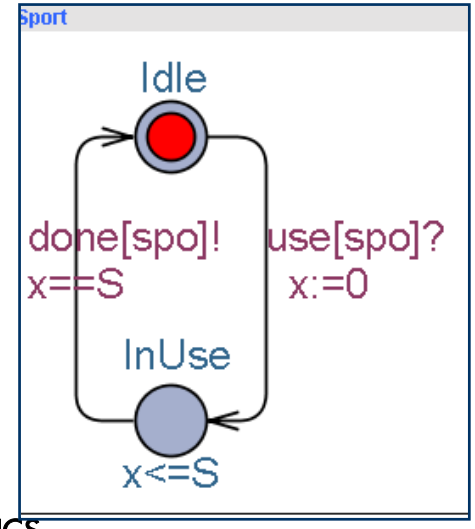
	Sport	Economy	Local News	Comic Stip
Kim	2. 5 min	4. 1 min	3. 3 min	1. 10 min
Maria	1. 10 min	2. 20 min	3. 1 min	4. 1 min
Nicola	4. 1 min	1. 13 min	3. 11 min	2. 11 min

Problem: compute the minimal **MAKESPAN**

BRICS
Basic Research
in Computer Science

Jobshop Scheduling in UPPAAL





	Sport	Economy	Local News	Comic Stip
Kim	2. 5 min	4. 1 min	3. 3 min	1. 10 min
Maria	1. 10 min	2. 20 min	3. 1 min	4. 1 min
Nicola	4. 1 min	1. 13 min	3. 11 min	2. 11 min

BRICS
Basic Research
in Computer Science

Experiments

[TACAS'2001]

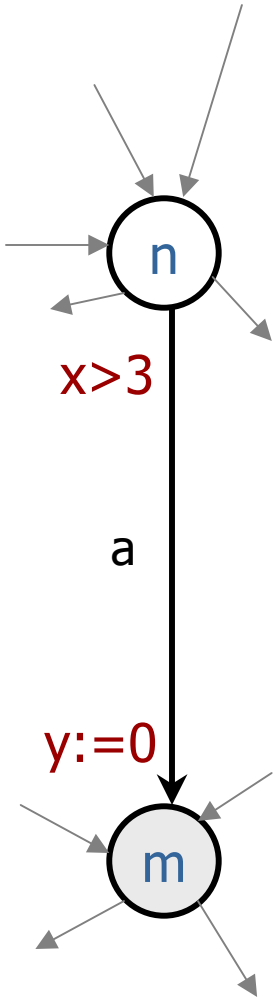
B-&-B algorithm running for 60 sec.

		problem	BF		MC		MC+		DF		RDF		comb. heur.		minimal
		instance	cost	states	cost	states	cost	states	cost	states	cost	states	cost	states	makespan
m=5	j=10	la01	-	-	-	-	-	-	2466	-	842	-	666	292	666
		la02	-	-	-	-	-	-	2360	-	806	-	672	-	655
		la03	-	-	-	-	-	-	2094	-	769	-	626	-	597
		la04	-	-	-	-	-	-	2212	-	783	-	639	-	590
		la05	-	-	-	-	593	9791	1955	-	696	-	593	284	593
	j=15	la06	-	-	-	-	-	-	3656	-	1076	-	926	480	926
		la07	-	-	-	-	-	-	3410	-	1113	-	890	-	890
		la08	-	-	-	-	-	-	3520	-	1009	-	863	400	863
		la09	-	-	-	-	-	-	3984	-	1154	-	951	425	951
		la10	-	-	-	-	-	-	3681	-	1063	-	958	454	958
	j=20	la11	-	-	-	-	-	-	4974	-	1303	-	1222	642	1222
		la12	-	-	-	-	-	-	4557	-	1271	-	1039	633	1039
		la13	-	-	-	-	-	-	4846	-	1227	-	1150	662	1150
		la14	-	-	-	-	1292	10653	5145	-	1377	-	1292	688	1292
		la15	-	-	-	-	-	-	5264	-	1459	-	1289	-	1207
m=10	j=10	la16	-	-	-	-	-	-	4849	-	1298	-	1022	-	945
		la17	-	-	-	-	-	-	4299	-	938	-	786	-	784
		la18	-	-	-	-	-	-	4763	-	1034	-	922	-	848
		la19	-	-	-	-	-	-	4566	-	1140	-	904	-	842
		la20	-	-	-	-	-	-	5056	-	1378	-	964	-	902
	j=15	la21	-	-	-	-	-	-	7608	-	1326	-	1149	-	(1040,1053)
		la22	-	-	-	-	-	-	6920	-	1413	-	1047	-	927
		la23	-	-	-	-	-	-	7676	-	1357	-	1075	-	1032
		la24	-	-	-	-	-	-	7237	-	1346	-	1061	-	935
		la25	-	-	-	-	-	-	7141	-	1290	-	1070	-	977

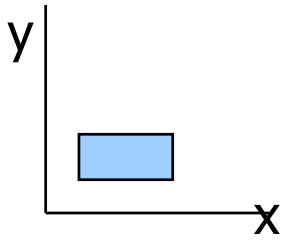
Lawrence Job Shop Problems

Symbolic Transitions

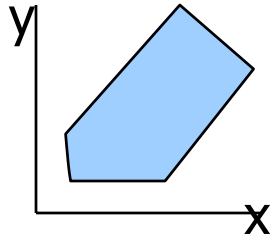
using Zones



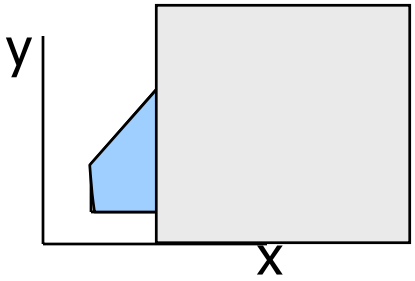
1<=x<=4
1<=y<=3



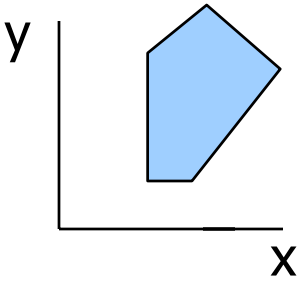
delays to



1<=x, 1<=y
-2<=x-y<=3

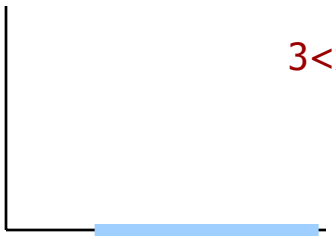


conjuncts to



3<x, 1<=y
-2<=x-y<=3

projects to



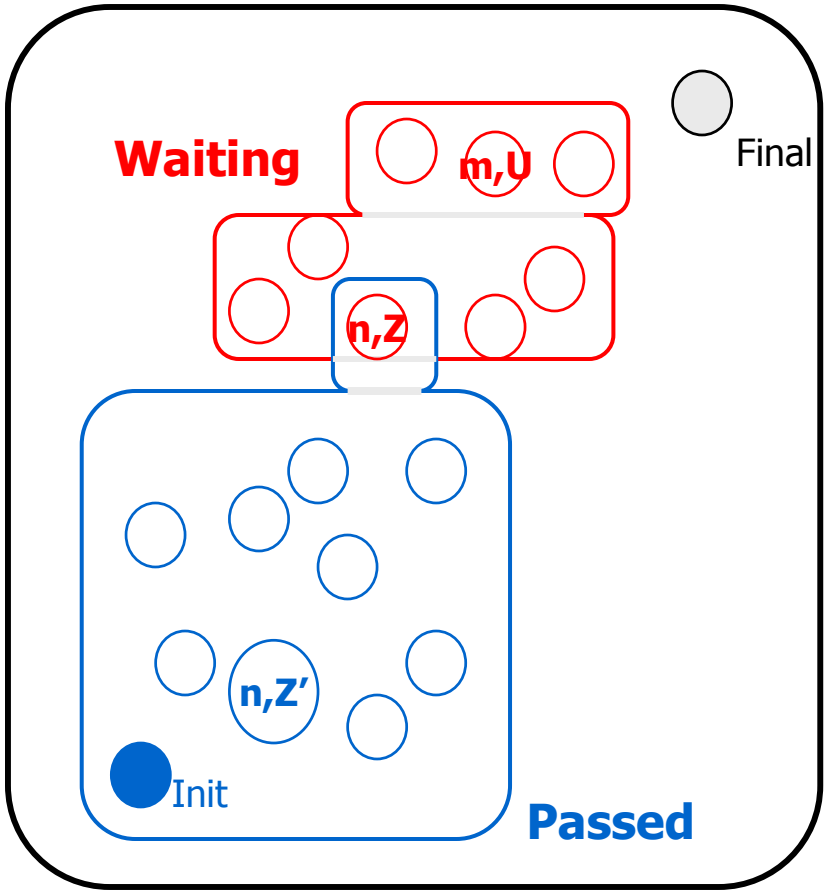
3<x, y=0

Thus (n,1<=x<=4,1<=y<=3) =a => (m,3<x, y=0)

BRICS
Basic Research
in Computer Science

Forward Rechability

Init -> Final ?



INITIAL **Passed** := Ø;
Waiting := {(n0,Z0)}

REPEAT
- pick (n,Z) in **Waiting**
- **if** for some Z' ⊇ Z
(n,Z') in **Passed** **then** **STOP**
- **else** /explore/ add
{ (m,U) : (n,Z) => (m,U) }
to **Waiting**;
Add (n,Z) to **Passed**

UNTIL **Waiting** = Ø
or
Final is in **Waiting**

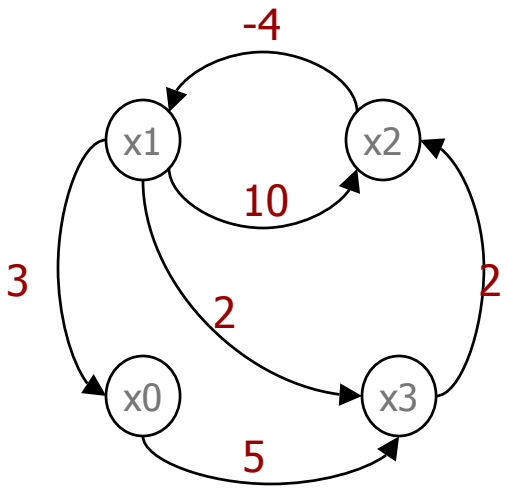
BRICS
Basic Research
in Computer Science

Canonical Datastructures for Zones

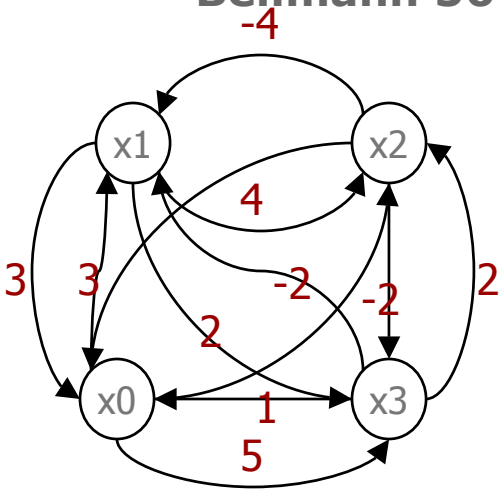
DBMs

Bellmann 56

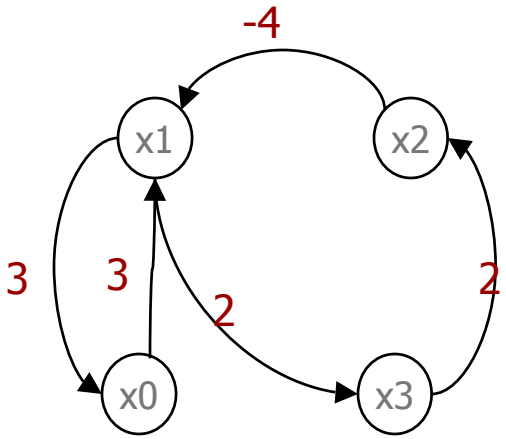
x1-x2<=4
x2-x1<=10
x3-x1<=2
x2-x3<=2
x0-x1<=3
x3-x0<=5



Shortest Path Closure O(n^3)



Shortest Path Reduction O(n^3)



Space worst O(n^2)
practice O(n)

Minimal Constraint Form

RTSS'97

BRICS
Basic Research
in Computer Science

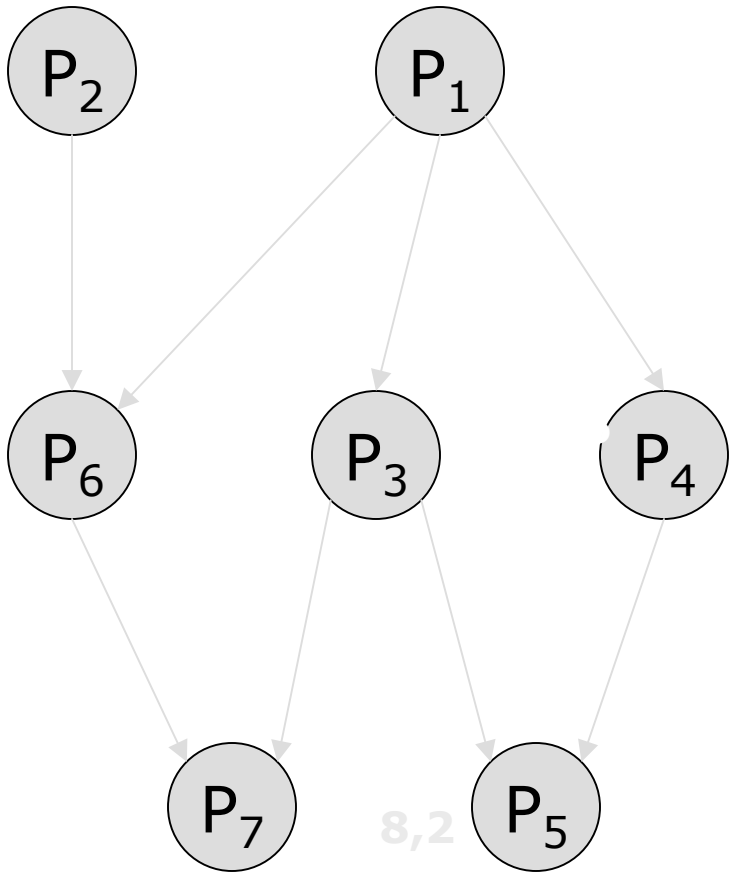
Datastructures for Zones

- DBM package
- M
  Fo
- Cl
  D

BRICS
Basic Research
in Computer Science

Task Graph Scheduling

Optimal Static Task Scheduling



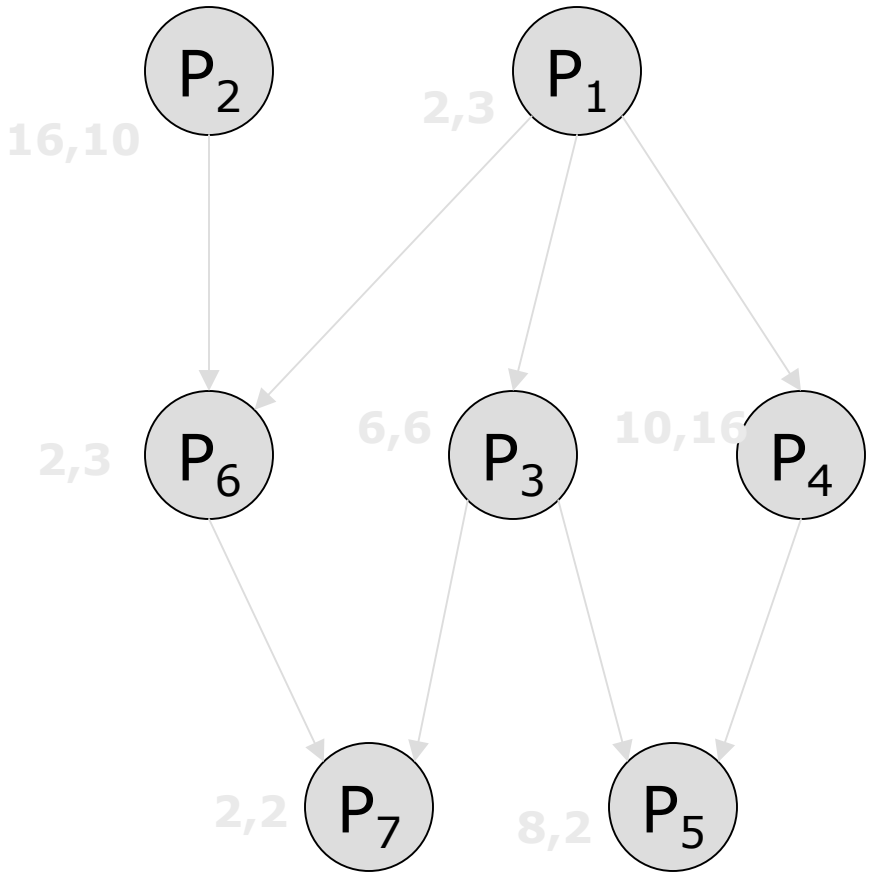
$\mathbf{M} = \{M_1, M_2\}$

BRICS
Basic Research
in Computer Science

Task Graph Scheduling

Optimal Static Task Scheduling

- Task **P**={P_1,.., P_m}
- Machines **M**={M_1,..,M_n}



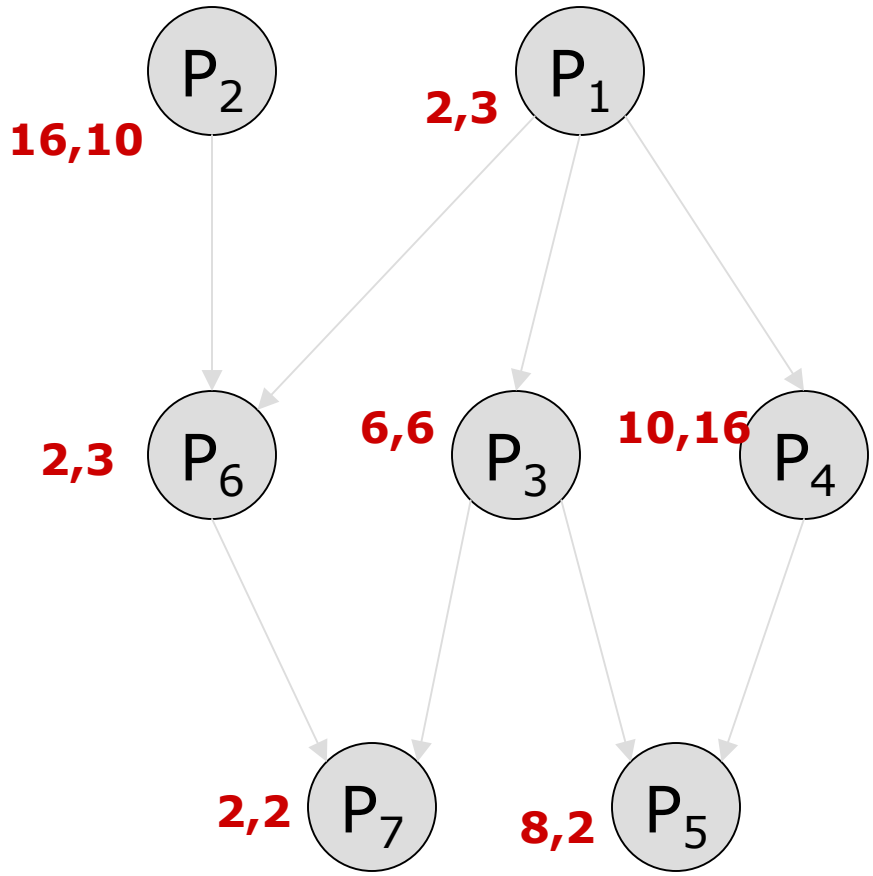
M = {M_1,M_2}

BRICS
Basic Research
in Computer Science

Task Graph Scheduling

Optimal Static Task Scheduling

- Task **P**=$\{P_1,..,P_m\}$
- Machines **M**=$\{M_1,..,M_n\}$
- Duration Δ : **(P**×**M)** → $\mathbf{N}_\infty$



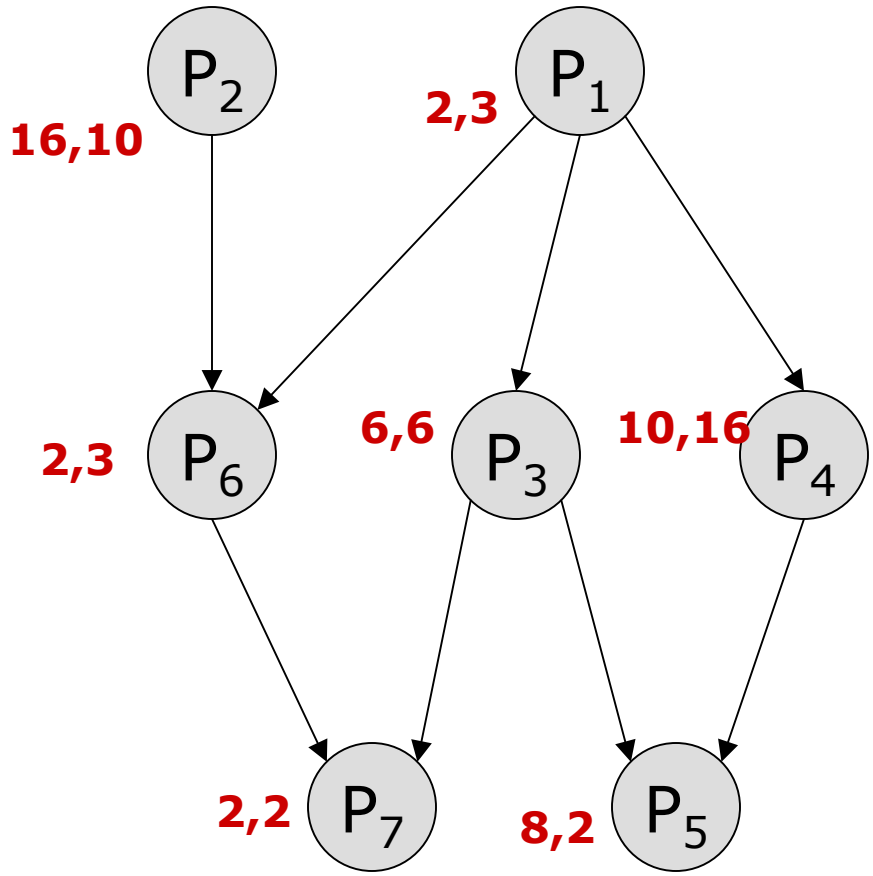
M = $\{M_1,M_2\}$

BRICS
Basic Research
in Computer Science

Task Graph Scheduling

Optimal Static Task Scheduling

- Task **P**=$\{P_1,..,P_m\}$
- Machines **M**=$\{M_1,..,M_n\}$
- Duration Δ : **(P**×**M)** → $\mathbf{N}_\infty$
- < : p.o. on **P** (pred.)



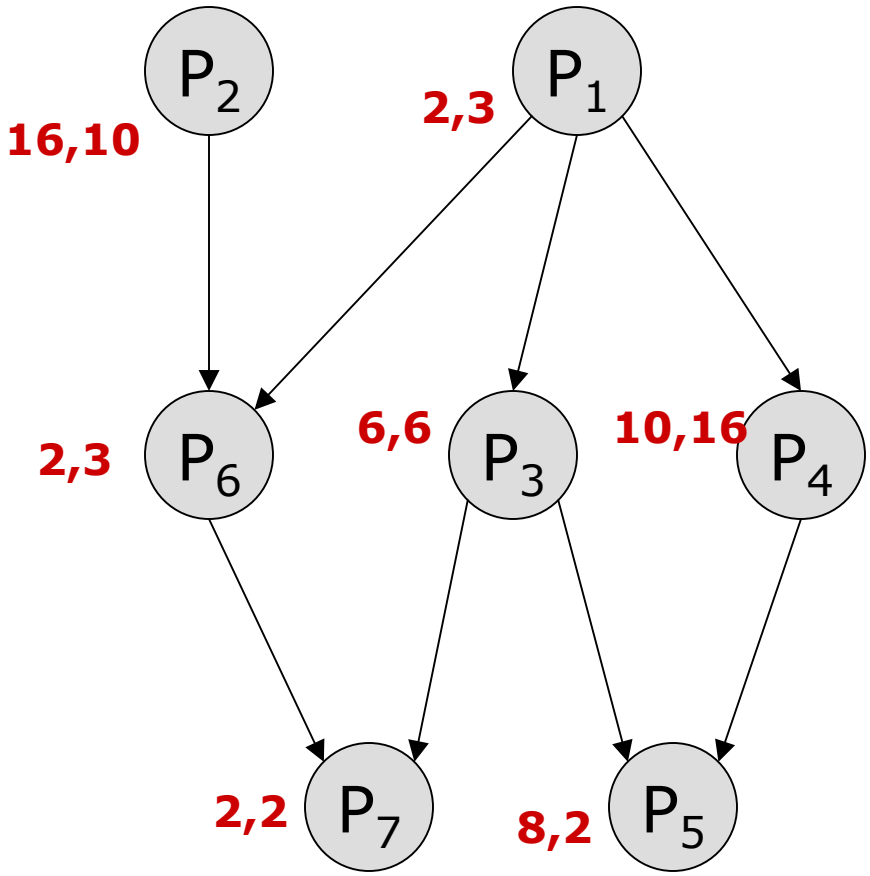
M = $\{M_1,M_2\}$

BRICS
Basic Research
in Computer Science

Task Graph Scheduling

Optimal Static Task Scheduling

- Task **P**=$\{P_1,..,P_m\}$
- Machines **M**=$\{M_1,..,M_n\}$
- Duration Δ : **(P**×**M)** → $\mathbf{N}_\infty$
- < : p.o. on **P** (pred.)



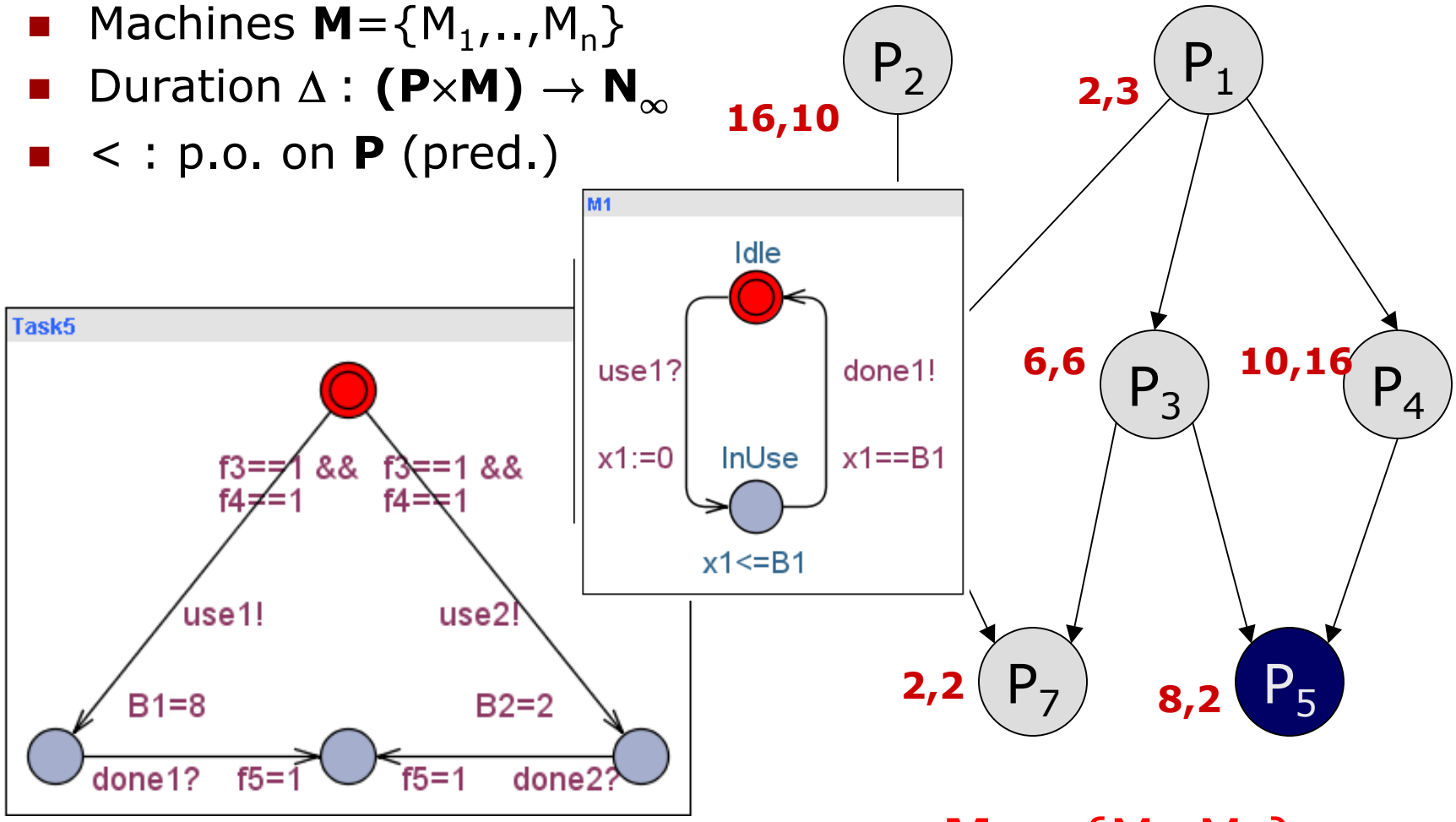
- A task can be executed only if all predecessors have completed
- Each machine can process at most one task at a time
- Task cannot be preempted.

M = $\{M_1,M_2\}$

Task Graph Scheduling

Optimal Static Task Scheduling

- Task **P**={P_1,.., P_m}
- Machines **M**={M_1,..,M_n}
- Duration Δ : **(P**×**M)** → $\mathbf{N}_\infty$
- < : p.o. on **P** (pred.)



M = {M_1,M_2}

BRICS Basic Research in Computer Science

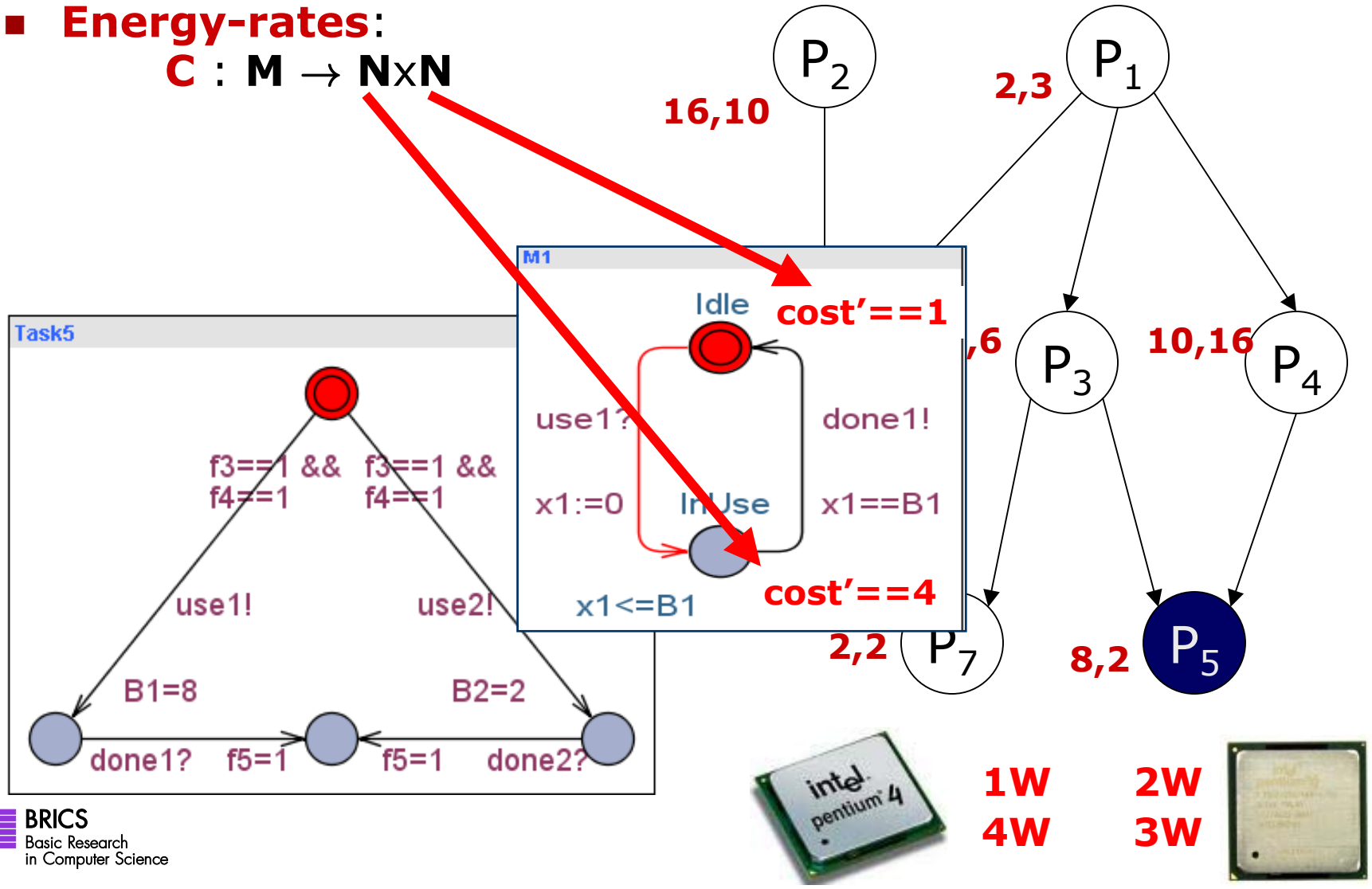
Experimental Results

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

Optimal Task Graph Scheduling

Power-Optimality

- **Energy-rates**:
  C : M → NxN



BRICS
Basic Research
in Computer Science

Overview

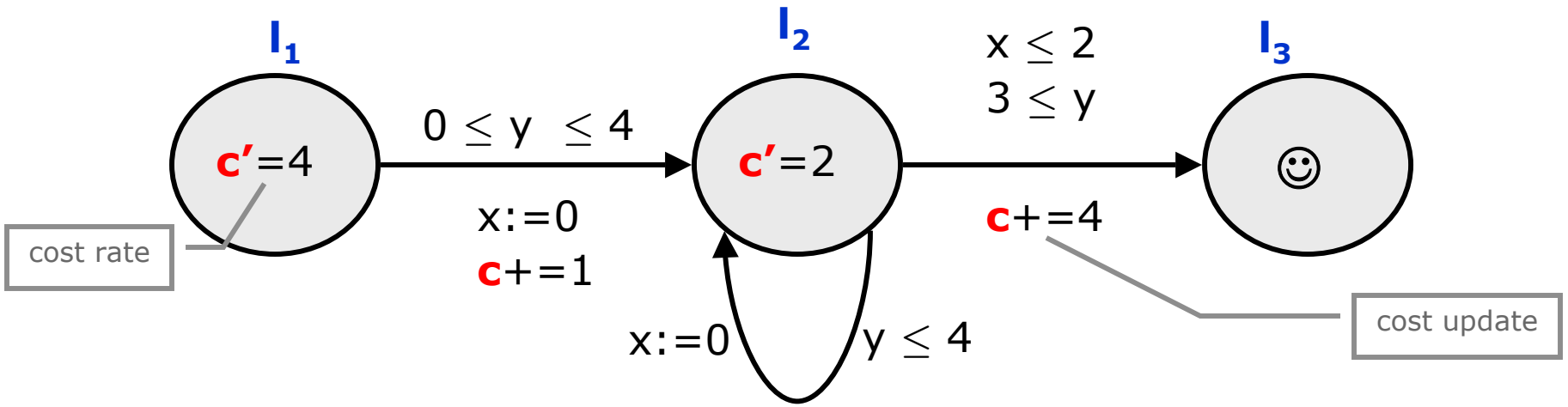
- Timed Automata & Scheduling
- Priced Timed Automata and Optimal Scheduling
- AMETIST Results
- Beyond Static Scheduling

Priced Timed Automata

Timed Automata + **COST** variable

Behrmann, Fehnker, et all (HSCC'01)

Alur, Torre, Pappas (HSCC'01)

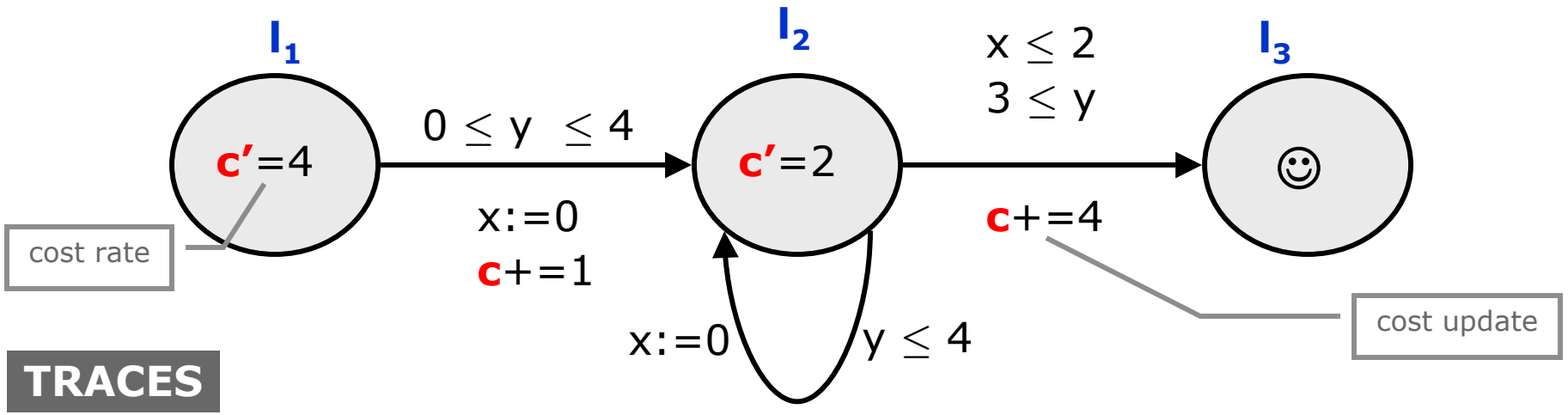


Priced Timed Automata

Timed Automata + **COST** variable

Behrmann, Fehnker, et all (HSCC'01)

Alur, Torre, Pappas (HSCC'01)



TRACES

$(l_1, x=y=0) \xrightarrow[12]{\varepsilon(3)} (l_1, x=y=3) \xrightarrow{1} (l_2, x=0, y=3) \xrightarrow{4} (l_3, \_, \_)$

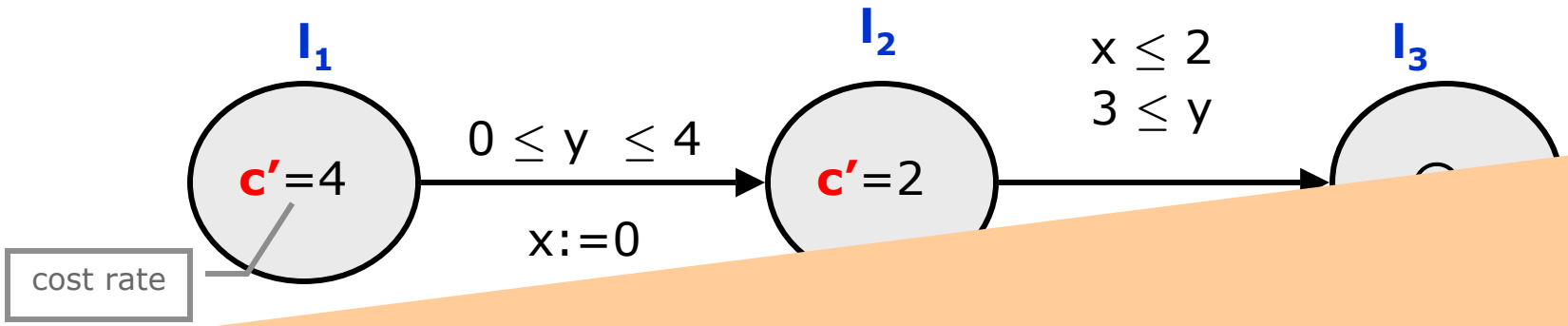
Σ c=17

Priced Timed Automata

Timed Automata + **COST** variable

Behrmann, Fehnker, et all (HSCC'01)

Alur, Torre, Pappas (HSCC'01)



Problem :

Find the minimum cost of reaching location l_3

Efficient Implementation:
CAV'01 and TACAS'04

$(l_1, x=y=0) \xrightarrow{1} (l_2, x=0, y=0) \xrightarrow{\varepsilon(3)\ 6} (l_2, x=3, y=3) \xrightarrow{0} (l_2, x=0, y=3) \xrightarrow{4} (l_3, \_, \_)$

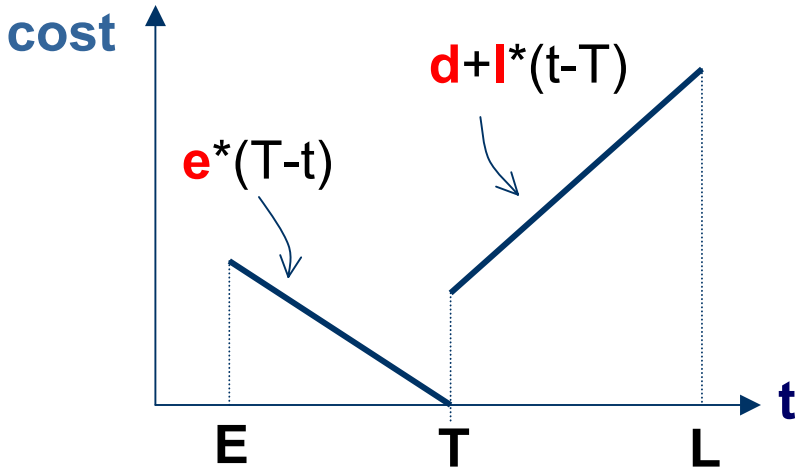
Σ c=11

$(l_3, \_, \_)$

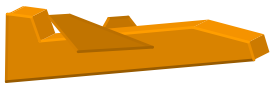
Σ c=16

BRICS
Basic Research
in Computer Science

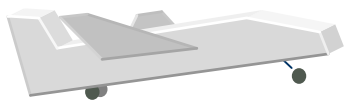
Aircraft Landing Problem

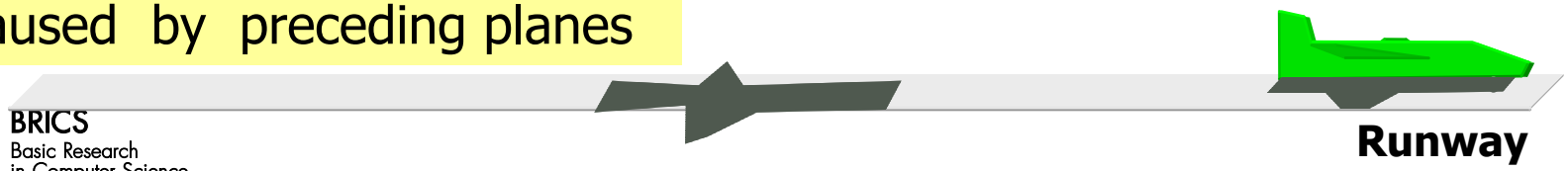


E earliest landing time
T target time
L latest time
e cost rate for being early
l cost rate for being late
d fixed cost for being late



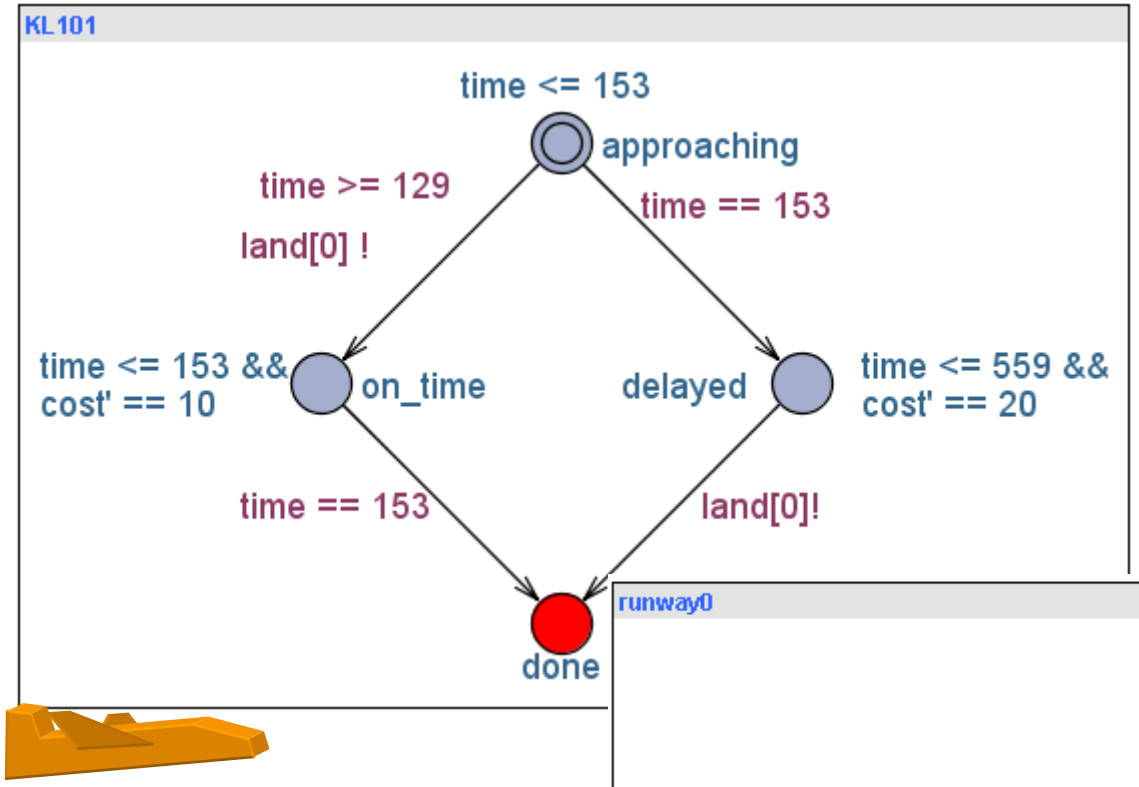
Planes have to keep separation distance to avoid turbulences caused by preceding planes





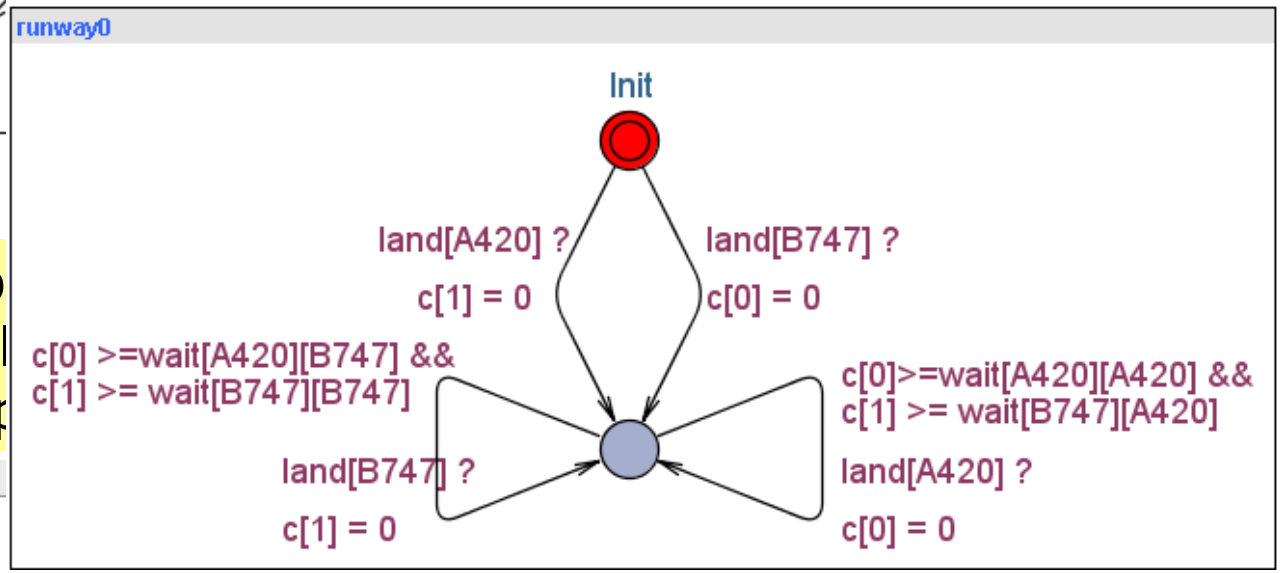
BRICS
Basic Research
in Computer Science

Modeling ALP with PTA



129: Earliest landing time
153: Target landing time
559: Latest landing time
10: Cost rate for early
20: Cost rate for late

Runway handles 2 types of planes



Planes have to keep sep
distance to avoid turbul
caused by preceding p

BRICS
Basic Research
in Computer Science

Symbolic ”A*”

Zones

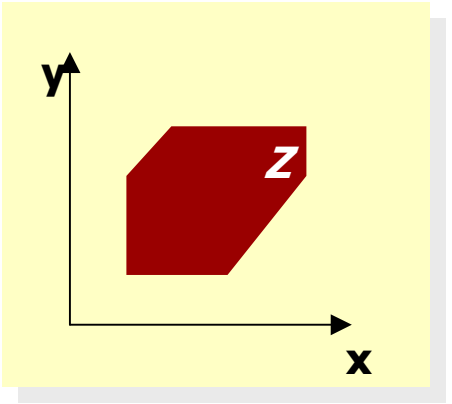
Definition

A *zone* Z over a set of clocks C is a finite conjunction of simple constraints of the forms:

$$x \geq l \qquad x \leq u \qquad x - y \geq l' \qquad x - y \leq u'$$

where $x, y \in C$, $l, u \in \mathbb{N}$ and $l', u' \in \mathbb{Z}$.

For $u \in \mathbb{R}^C$ and Z a zone we write $u \models Z$ if u satisfies all constraints of Z.



Operations

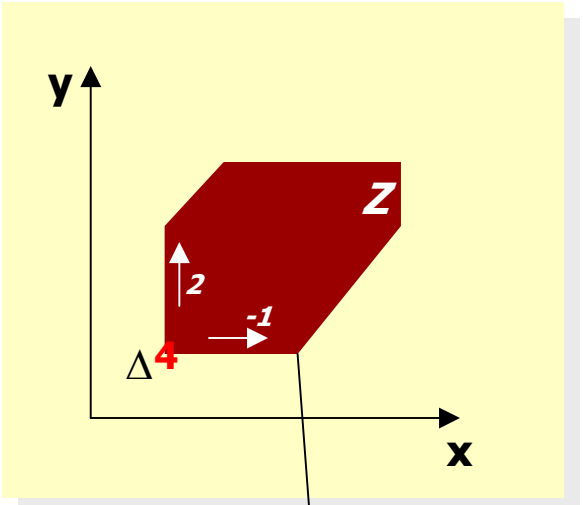
Reset: $\{x\}Z = \{u[0/x] \mid u \models Z\}$

Delay: $Z^{\uparrow} = \{u + d \mid u \models Z\}$

Offset: $\Delta_Z \models Z$ such that $\forall u \models Z. \forall x \in C.\, \Delta_Z(x) \leq u(x)$.

Priced Zone

CAV'01



Definition

A priced zone P is a tuple (Z, c, r), where:

- Z is a zone
- $c \in \mathbb{N}$ describes the cost of Δ_Z
- $r : C \to \mathbb{Z}$ gives a *rate* for any clock $x \in C$.

We write $u \models P$ whenever $u \models Z$. For $u \models P$ we define $\mathrm{Cost}(u, P)$ as follows:

$$\mathrm{Cost}(u, P) = c + \sum_{x \in C} r(x) \cdot (u(x) - \Delta_Z(x))$$

$$Cost(x, y) = 2y - x + 2$$

Branch & Bound Algorithm

Cost := ∞
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) + $\text{Rem}_{(l,Z)} \geq$ 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 Waiting
return Cost

Branch & Bound Algorithm

Cost := ∞
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) + $\text{Rem}_{(l,Z)} \geq$ 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 Waiting
return Cost

Branch & Bound Algorithm

Cost := ∞
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) + $\text{Rem}_{(l,Z)} \geq$ 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 Waiting
return Cost

Branch & Bound Algorithm

Cost := ∞
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 Waiting
return Cost

Branch & Bound Algorithm

Cost := ∞
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 Waiting
return Cost

Branch & Bound Algorithm

Cost := ∞
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$_{(l,Z)} \geq$ C… **th**…
  if for all (l, Z') in Passed: $Z' \not\leq Z$ **then**
    add (l, Z) to Passed
    add all (l', Z') with $(l, Z) \rightarrow (l',$

return Cost

$Z' \leq Z$

Z′* is bigger & cheaper than *Z

≤ is a well-quasi ordering which guarantees termination!

Experimental Results

Modeling APL with PDDL$_3$

[Dierks VVPS05]

```
(define (domain Planes)
  (:requirements :typing :fluents :negative-preconditions
   :conditional-effects :equality :duration-inequalities
   :durative-actions)
  (:types plane runway)
  (:predicates  (occupied ?r - runway)
                (landed   ?p - plane)
                (scheduled ?p - plane ?r - runway))
  (:functions   (earliest     ?p - plane)
                (target       ?p - plane)
                (latest       ?p - plane)
                (early-rate   ?p - plane)
                (late-rate    ?p - plane)
                (late-penalty ?p - plane)
                (cost))
```

```
(:durative-action land-late     :parameters (?p - plane ?r - runway)
 :duration (<= ?duration (- (latest ?p) (earliest ?p)))
 :condition (and         (at end   (occupied ?r))
                         (at end   (scheduled ?p ?r))
                         (at start (= total-time (target ?p))))
 :effect (and (at end (landed ?p))
              (at start (increase cost (late-penalty ?p)))
              (increase cost (late-rate ?p))))
)
```

BRICS
Basic Research
in Computer Science

Modeling APL with PDDL$_3$

[Dierks VVPS05]

```
(define (problem plane1)
  (:domain Planes)
  (:objects     KL101,KL108,KL115 - plane
                r - runway)

  (:init (not (occupied r))
         (not (landed   KL101)) (not (scheduled KL101 r))
         (not (landed   KL108)) (not (scheduled KL108 r))
         (not (landed   KL115)) (not (scheduled KL115 r))

        (= (earliest KL101) 129) (= (early-rate   KL101) 10)
        (= (target   KL101) 155) (= (late-rate    KL101) 10)
        (= (latest   KL101) 559) (= (late-penalty KL101) 500)

        (= (earliest KL108) 195) (= (early-rate   KL108) 10)
        (= (target   KL108) 258) (= (late-rate    KL108) 10)
        (= (latest   KL108) 744) (= (late-penalty KL108) 500)

        (= (earliest KL115) 89)  (= (early-rate   KL115) 30)
        (= (target   KL115) 98)  (= (late-rate    KL115) 30)
        (= (latest   KL115) 510) (= (late-penalty KL115) 500)
  )

  (:goal (and   (landed KL101)
                (landed KL108)
                (landed KL115)
                (not (occupied r))))
  (:metric minimize cost)
)
```

Experimental Results

Translating $PDDL_3$ to PTA

[Dierks VVPS05+]

# planes	# inst	# clocks	costs	time (s)	mem (MB)
3	2	3	0	0.5	5
3	3	4	0	1.9	9
3	4	5	0	9	28
4	2	3	860	0.8	7
4	3	4	860	7.4	29
4	4	5	860	59	168
5	2	3	1540	2.4	16
5	3	4	1540	30	114
5	4	5	out of mem (400MB,97 s)		
6	2	3	180	10.3	56
6	3	4	60	out of mem (96 s)	
6	4	5	out of mem (76 s)		
7	2	3	920	30.8	166
7	3	4	out of mem (66 s)		
8	2	3	1870	out of mem (63 s)	
8	3	4	out of mem (60 s)		
9	2	3	out of mem (55 s)		

BRICS Basic Research in Computer Science

Experimental Results

PTA versus MILP

[Beasley'00]

RW	Planes	10	15	20	20	20	30	44
	Types	2	2	2	2	2	4	2
1	MILP (s)	*0.4*	*5.2*	*2.7*	*220.4*	*922.0*	*33.1*	*10.6*
	MC (s)	**0.8**	**5.6**	**2.8**	**20.9**	**49.9**	**0.6**	**2.2**
	Factor	*2.0*	*1.08*	*1.04*	**10.5**	**18.5**	**55.2**	**48.1**
2	MILP (s)	*0.6*	*1.8*	*3.8*	*1919.9*	*11510.4*	*1568.1*	*0.2*
	MC (s)	**2.7**	**9.6**	**3.9**	**138.5**	**187.1**	**6.0**	**0.9**
	Factor	*4.5*	*5.3*	*1.02*	**13.9**	**61.5**	**261.3**	*4.5*
3	MILP (s)	*0.1*	*0.1*	*0.2*	*2299.2*	*1655.3*	*0.2*	N/A
	MC (s)	**0.2**	**0.3**	**0.7**	**1765.6**	**1294.9**	**0.6**	
	Factor	*2.0*	*3.0*	*3.5*	**1.30**	**1.28**	*3.0*	
4	MILP (s)	N/A	N/A	N/A	*0.2*	*0.2*	N/A	N/A
	MC (s)				**3.3**	**0.7**		
	Factor				*16.5*	*3.5*		

BRICS
Basic Research
in Computer Science

Experimental Results

OPTOP PDDL$_3$

To be provided with guidance
from
Maria Fox & Drew McDermott

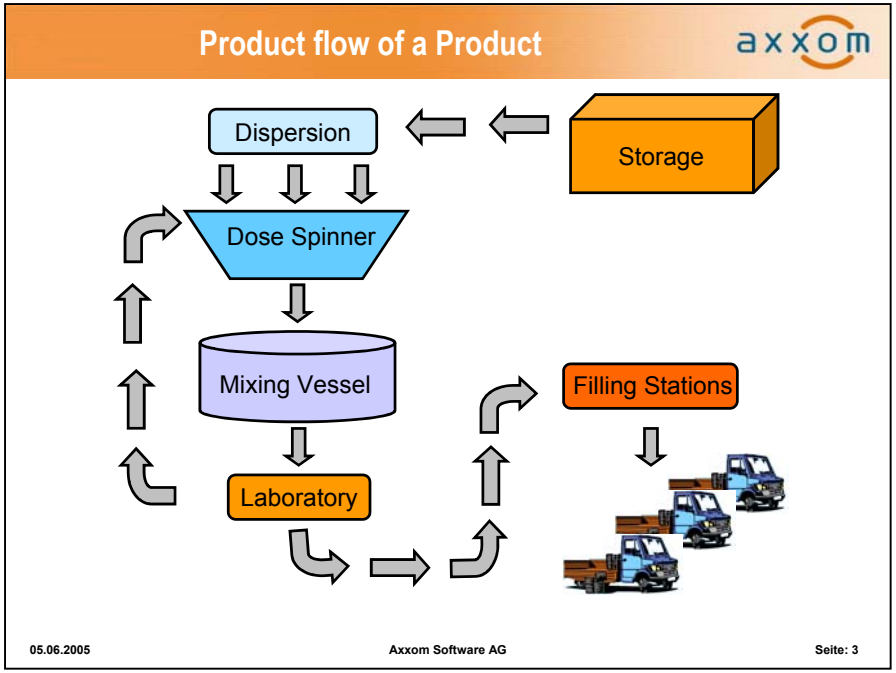
Overview

- Timed Automata & Scheduling
- Priced Timed Automata and Optimal Scheduling
- AMETIST Result
- Beyond Static Scheduling

AXXOM Case study

Laquer Production Scheduling

- 3 types of recipes
  - for uni/metallic/bronce
  - use of resources, processing times, timing
- 29 (73, 219) orders:
  - start time, due date, recipe
- extensions:
  - delay cost, storage cost, setup cost
  - weekend, nights

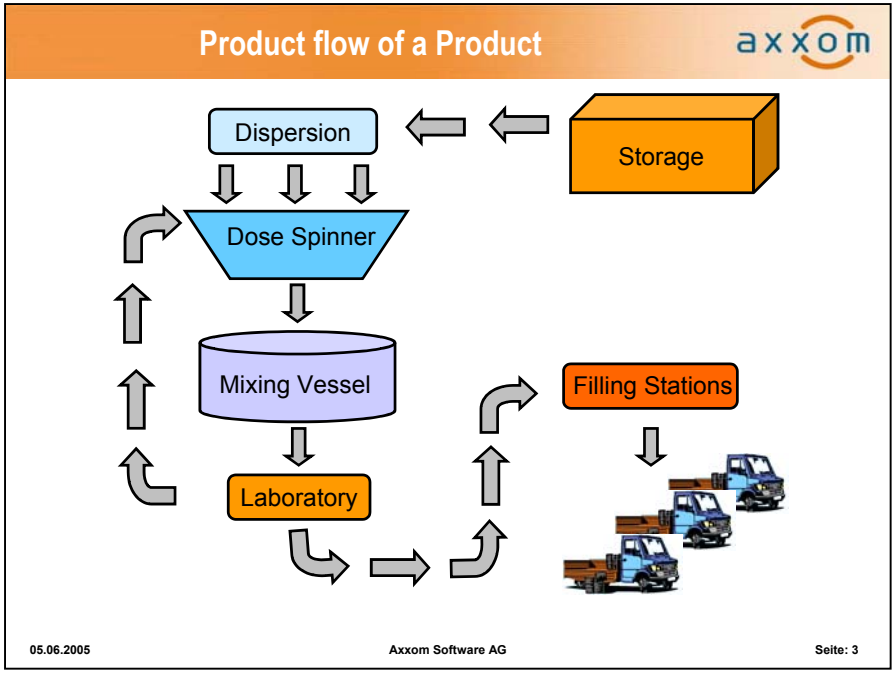


Behrmann, Brinksma, Hendriks, Mader
16th IFAC World Congress

BRICS
Basic Research
in Computer Science

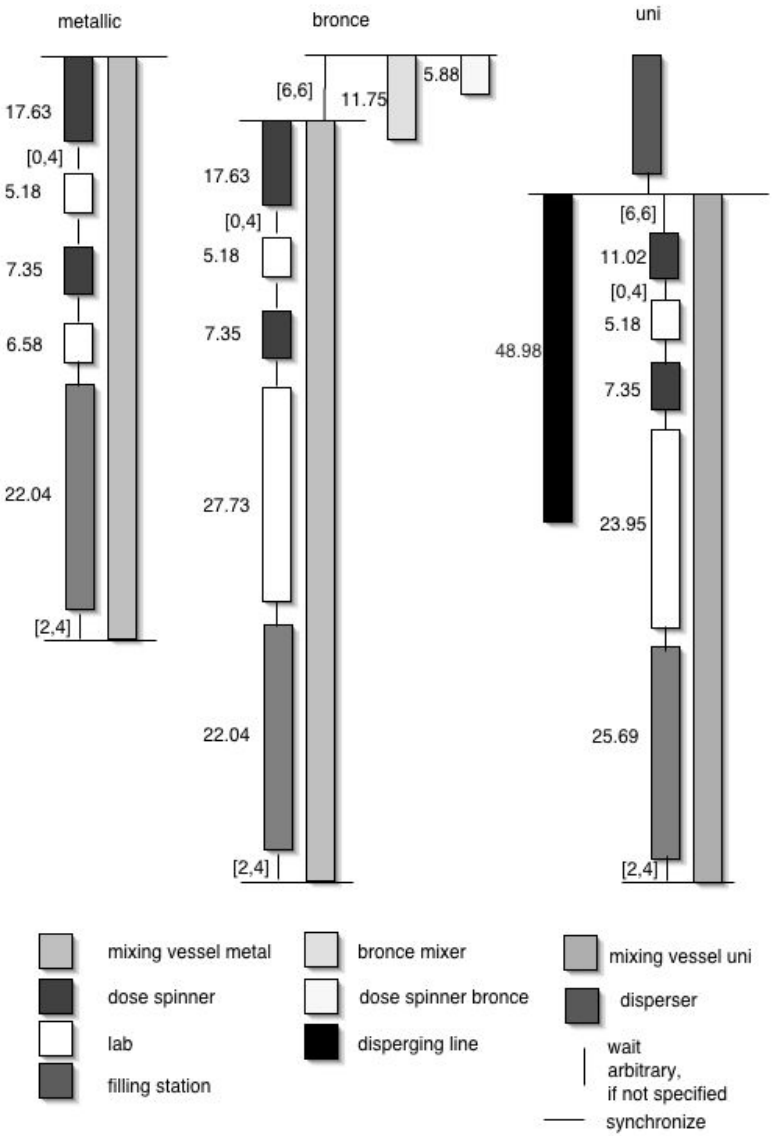
Resources

- 2 mixing vessels for uni lacquers
- 3 mixing vessels for metallic/bronce
- 2 dose spinners
- 1 dose spinner bronce
- 1 disperging line
- 1 predisperser
- 1 bronce mixer
- 2 filling lines
- lab (unlimitted)

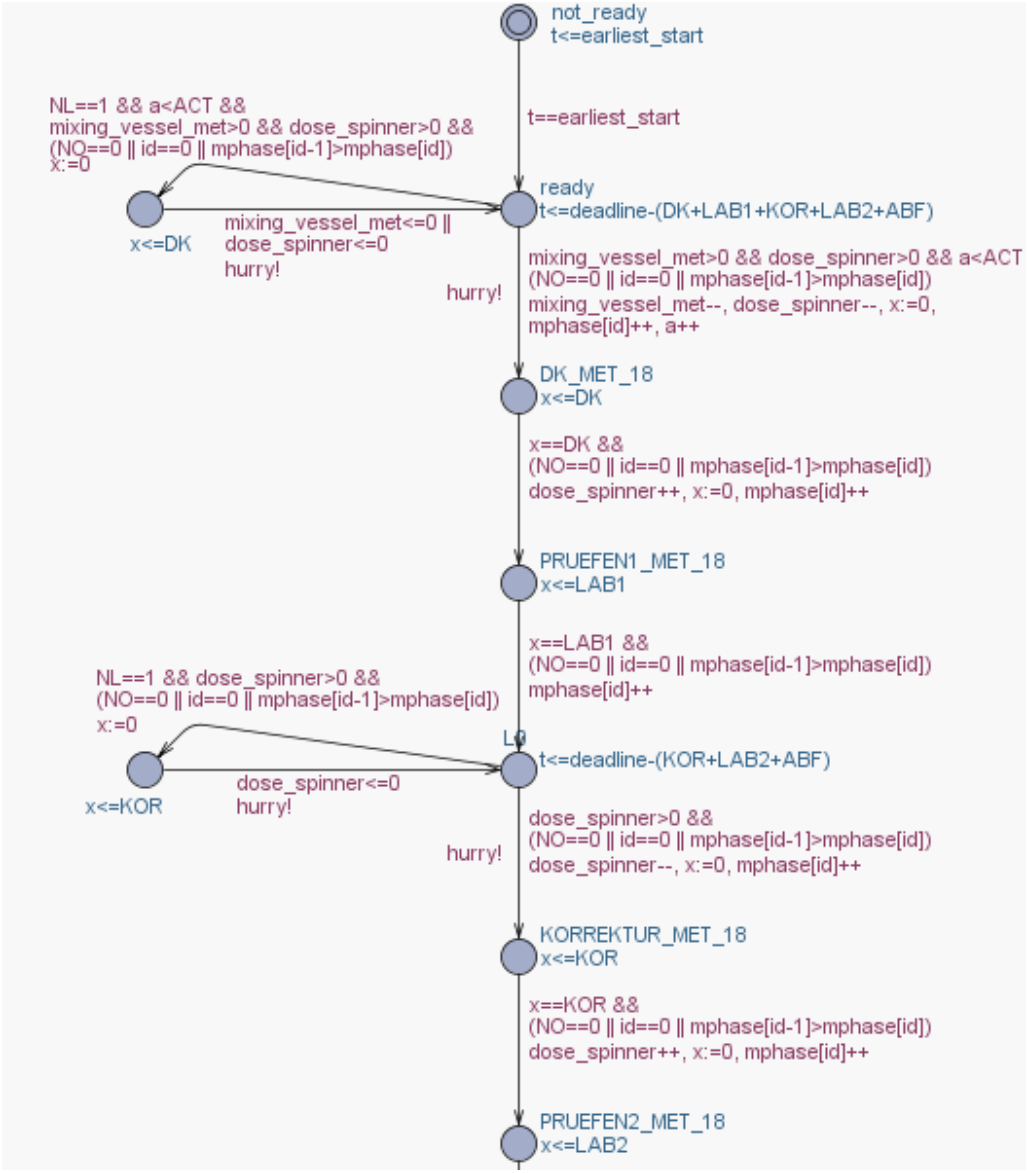


BRICS
Basic Research
in Computer Science

Recipes



UPPAAL template for metal



Orders

order	product ID	quantity	unit	delay costs per day per kg	earliest start	delivery date
1204081.FE	10009.ABF.MET.18	19000	kg	51	05.04.2002 19:12	19.04.2002 19:12
4147108.FE	10009.ABF.MET.18	19000	kg	51	20.04.2002 06:00	04.05.2002 06:00
4152856.FE	10009.ABF.MET.18	19000	kg	51	29.03.2002 07:12	12.04.2002 07:12
4161304.FE	10001.ABF.UNI.85	23000	kg	51	08.04.2002 04:00	22.04.2002 04:00
4173559.FE	10001.ABF.UNI.85	22000	kg	0,51	01.04.2002 22:48	15.04.2002 22:48
4173830.FE	10001.ABF.UNI.85	22000	kg	0,51	15.04.2002 02:48	29.04.2002 02:48
4174631.FE	10009.ABF.MET.18	19000	kg	51	22.04.2002 22:48	06.05.2002 22:48
4179351.FE	10009.ABF.MET.18	19000	kg	51	01.04.2002 23:59	15.04.2002 23:59
4179360.FE	10009.ABF.MET.18	19000	kg	51	29.03.2002 19:12	12.04.2002 19:12
4187241.FE	10001.ABF.UNI.85	22000	kg	0,51	15.04.2002 22:48	29.04.2002 22:48
4187676.FE	10009.ABF.MET.18	19000	kg	51	12.04.2002 19:12	26.04.2002 19:12
4187688.FE	10009.ABF.MET.18	19000	kg	51	12.04.2002 23:59	26.04.2002 23:59
4189165.FE	10009.ABF.MET.18	19000	kg	51	01.04.2002 04:00	15.04.2002 04:00
4189694.FE	10001.ABF.UNI.85	22000	kg	51	03.04.2002 10:24	17.04.2002 10:24
4196932.FE	10001.ABF.UNI.85	22000	kg	51	25.03.2002 04:39	08.04.2002 04:39
4196968.FE	10001.ABF.UNI.85	22000	kg	51	02.05.2002 14:24	16.05.2002 14:24
4197088.FE	10009.ABF.MET.18	19000	kg	51	27.03.2002 04:48	10.04.2002 04:48
4197955.FE	10001.ABF.UNI.85	22000	kg	51	03.04.2002 14:24	17.04.2002 14:24
4200741.FE	10009.ABF.MET.18	19000	kg	51	16.04.2002 04:48	30.04.2002 04:48
4206483.FE	10037.ABF.MET.11	19000	kg	51	15.04.2002 23:59	29.04.2002 23:59
4206683.FE	10037.ABF.MET.11	19000	kg	51	15.04.2002 04:00	29.04.2002 04:00
4209227.FE	10009.ABF.MET.18	19000	kg	51	12.04.2002 14:24	26.04.2002 14:24
4210646.FE	10001.ABF.UNI.85	22000	kg	51	26.04.2002 14:24	10.05.2002 14:24
4210669.FE	10037.ABF.MET.11	19000	kg	51	08.04.2002 04:00	22.04.2002 04:00
4213014.FE	10009.ABF.MET.18	19000	kg	51	23.03.2002 02:24	06.04.2002 02:24
4213016.FE	10009.ABF.MET.18	19000	kg	51	20.04.2002 02:24	04.05.2002 02:24
4220296.FE	10009.ABF.MET.18	19000	kg	51	27.03.2002 09:36	10.04.2002 09:36
4222941.FE	10037.ABF.MET.11	19000	kg	0,51	03.04.2002 06:00	17.04.2002 06:00
4223024.FE	10037.ABF.MET.11	19000	kg	0,51	18.04.2002 06:00	02.05.2002 06:00

Instantiated Model

Heuristics

- **Nice heuristics**
  - non-overtaking

    orders of the same recipe cannot overtake each other
  - non-laziness

    a process that needs an available resource will not waste time if its is not claimed by others (a.k.a. active scheduling)
- **Cut-and-Pray heuristics**
  - greediness

    a process that needs an available resource will claim this resource immediately
  - reducing active orders

    the number of concurrent orders is restricted (number of critical resources can give an indication)

Experimental Results

#jobs	heuristic	max. orders	term. time
29	-	-	-
29	nl	-	1 s
73	nl, no	-	-
73	nl, no	3	7 s
219	g, no	4	8 s

uses clock optimization & optimized successor calculation

Results Extended Case

storage, delay and setup costs, working hours

#jobs	work	ist	ma	
29				
2				
			-	14,875
2		o, g	-	2,263,496
29	ex	no	4	192,881,129

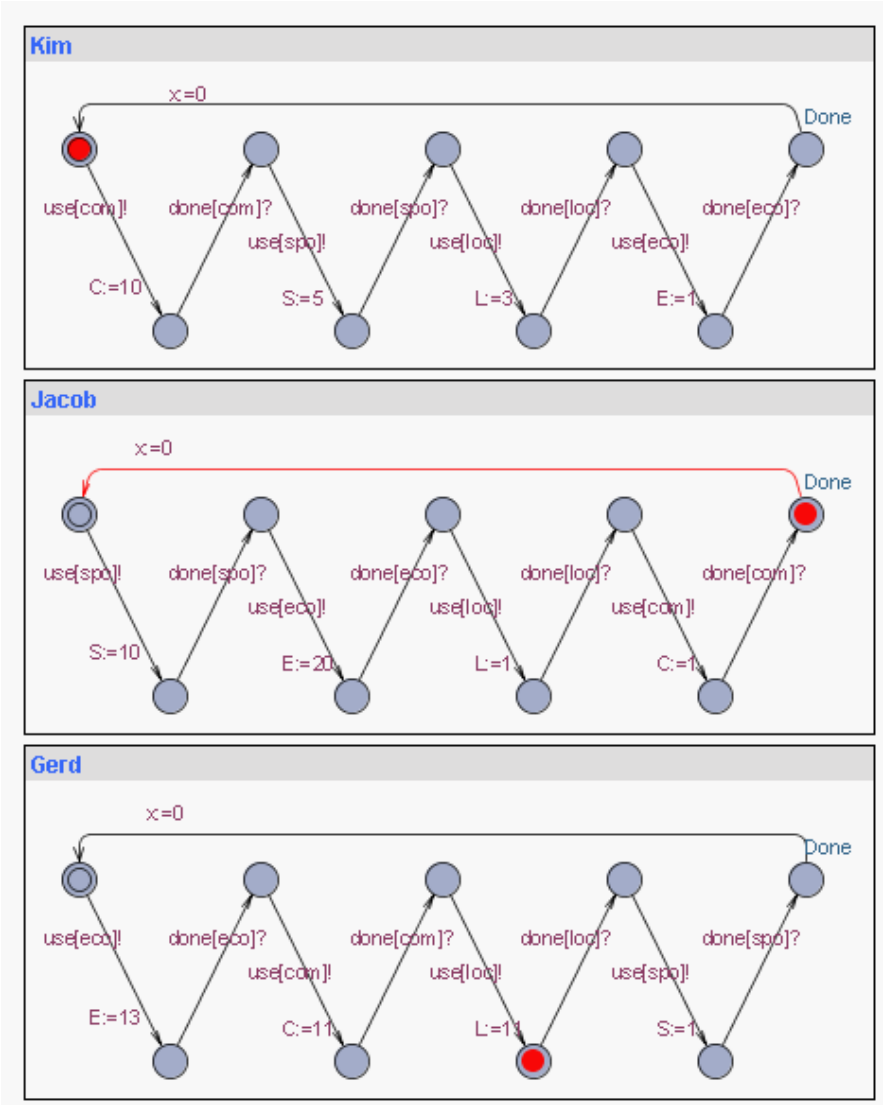
Competitive with Orion-pi results

Order of magnitude faster than MILP, GAMS/CPLEX

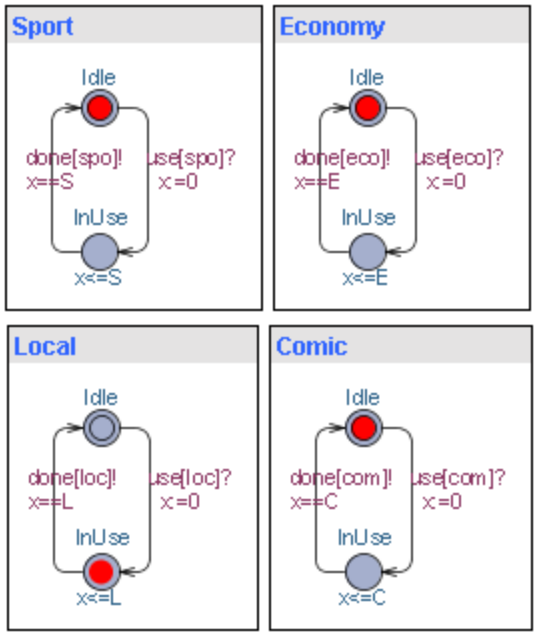
Overview

- Timed Automata & Scheduling
- Priced Timed Automata and Optimal Scheduling
- AMETIST Result
- Beyond Static Scheduling

Infinite Scheduling

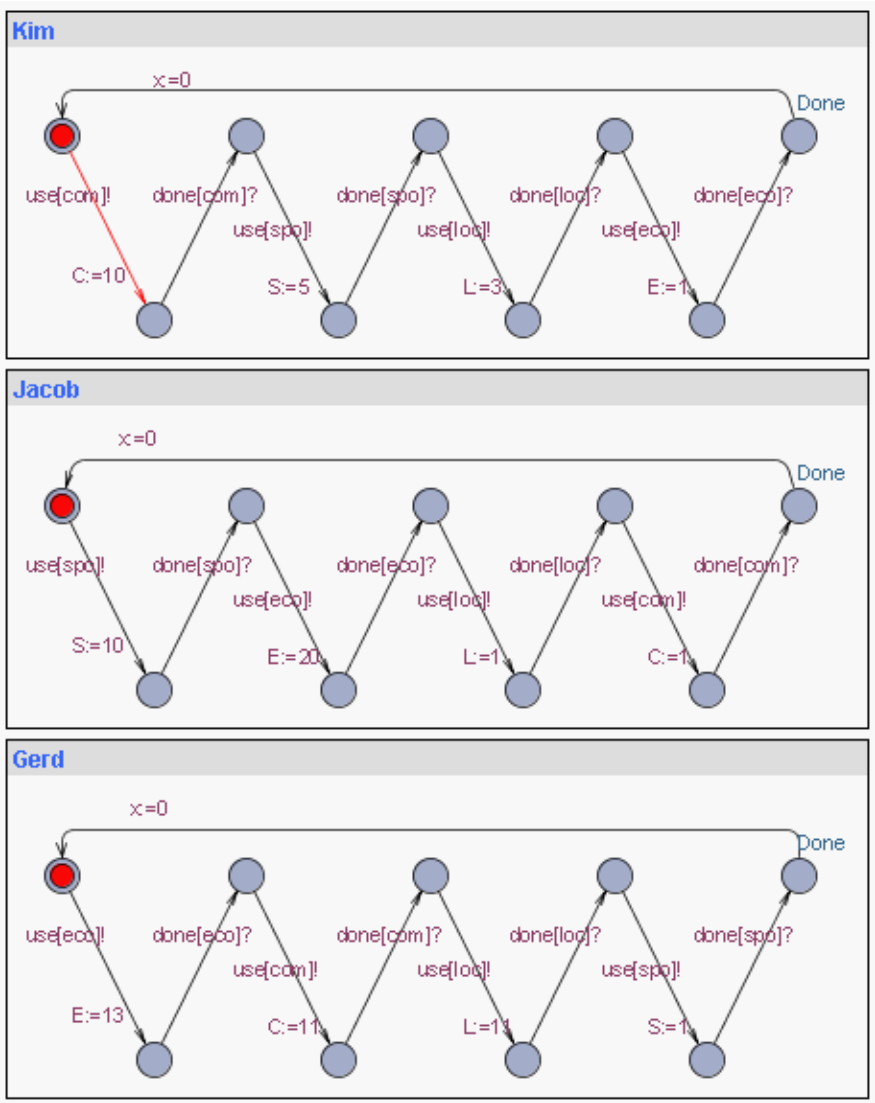


E[] (Kim.x ≤ 100 and
Jacob.x ≤ 90 and
Gerd.x ≤ 100)



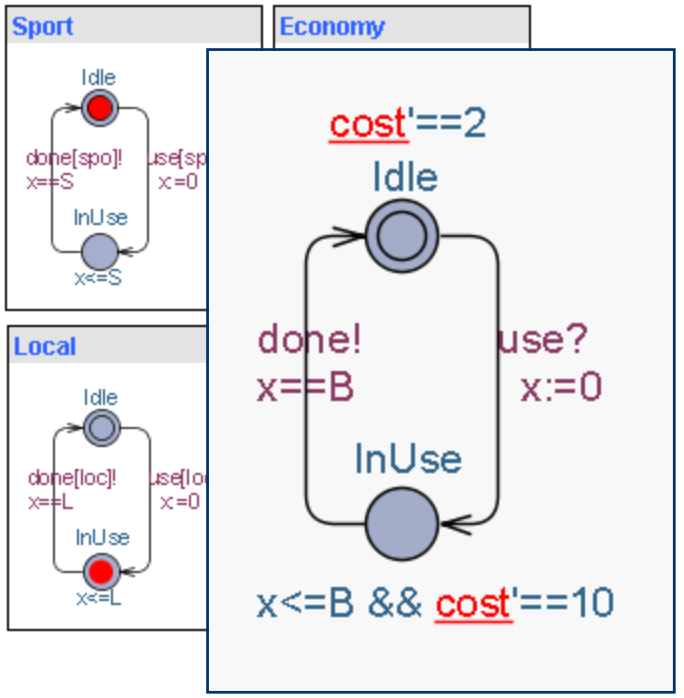
BRICS
Basic Research
in Computer Science

Infinite Optimal Scheduling



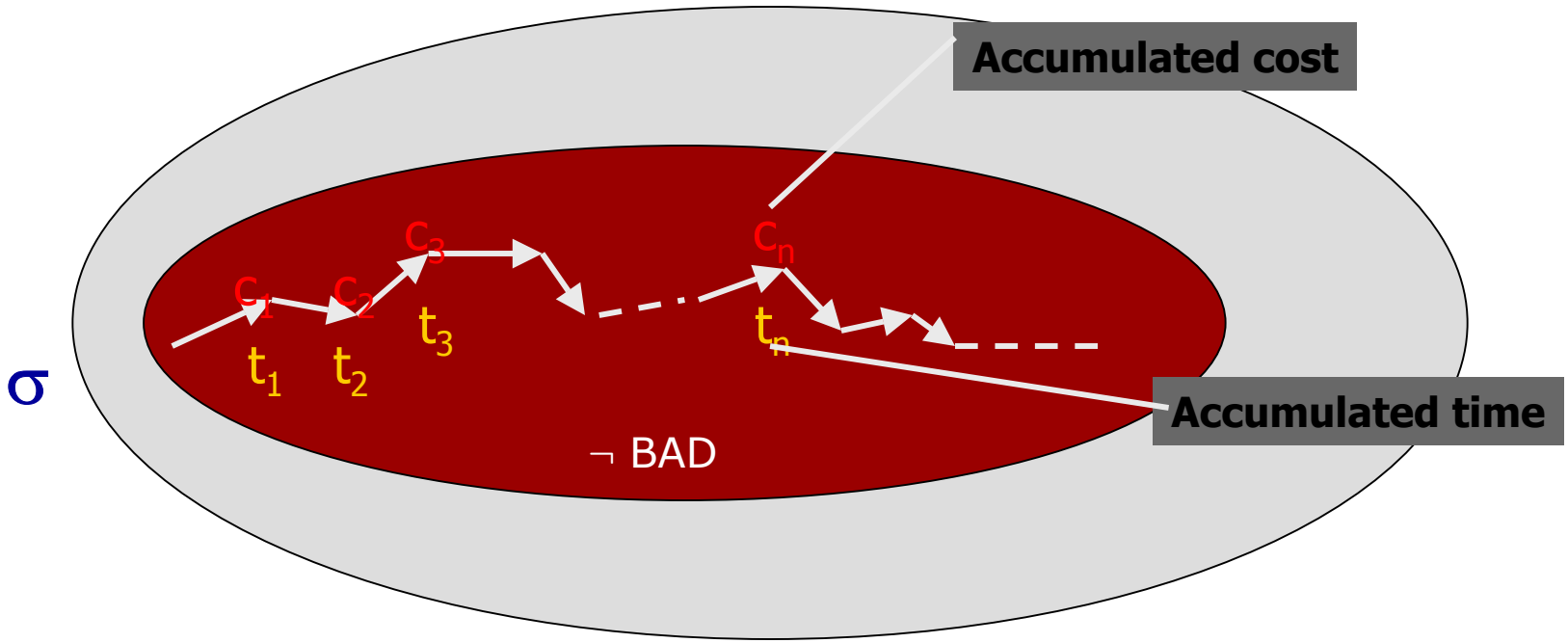
E[] (Kim.x ≤ 100 and
Jacob.x ≤ 90 and
Gerd.x ≤ 100)

+ most minimal limit of cost/time



BRICS
Basic Research
in Computer Science

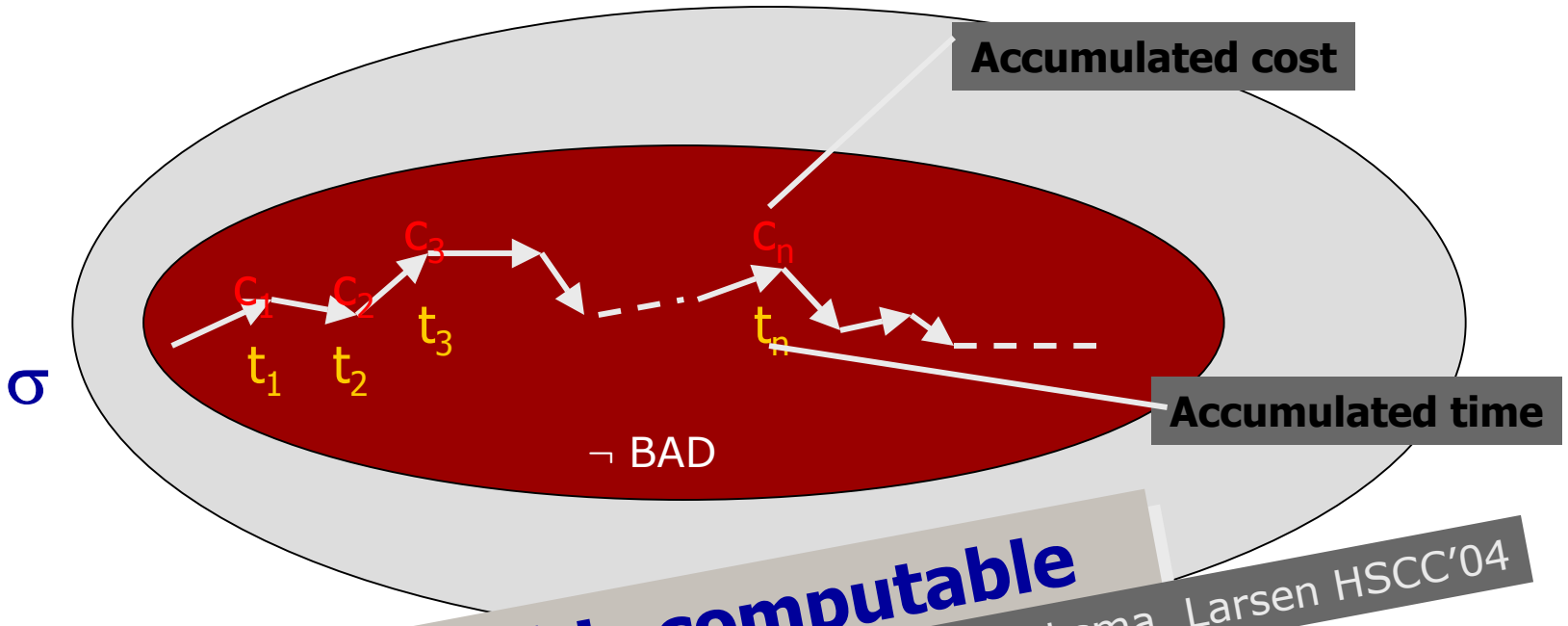
Infinite Optimal Scheduling



Value of path σ: $val(\sigma) = \lim_{n\to\infty} c_n/t_n$

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

Infinite Optimal Scheduling



THEOREM: σ^* is computable

Bouyer, Brinksma, Larsen HSCC'04

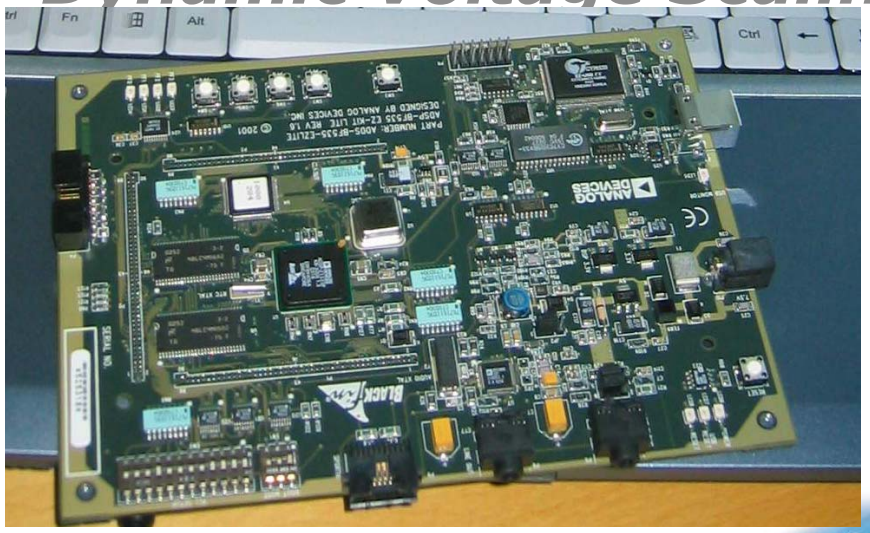
path σ: $val(\sigma) = \lim_{n\to\infty} c_n/t_n$

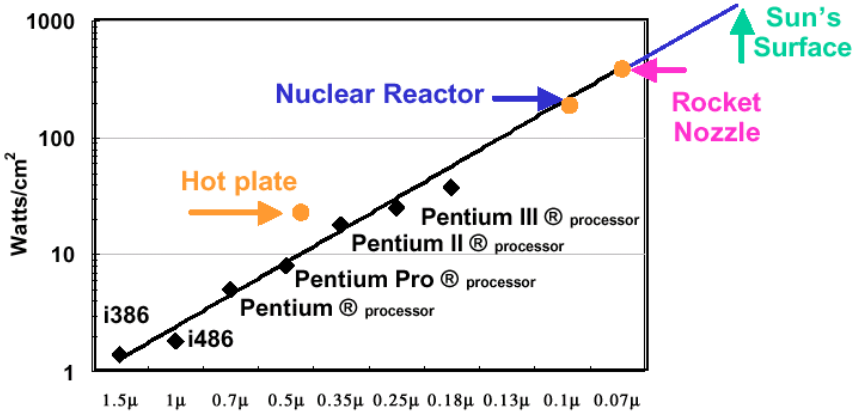
Optimal Schedule σ^*: $val(\sigma^*) = \inf_\sigma val(\sigma)$

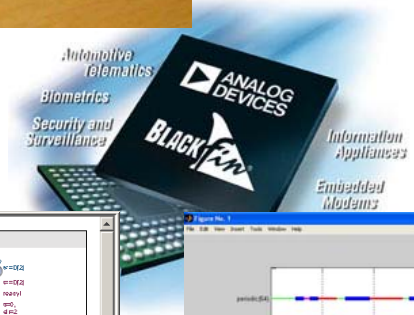
BRICS Basic Research in Computer Science

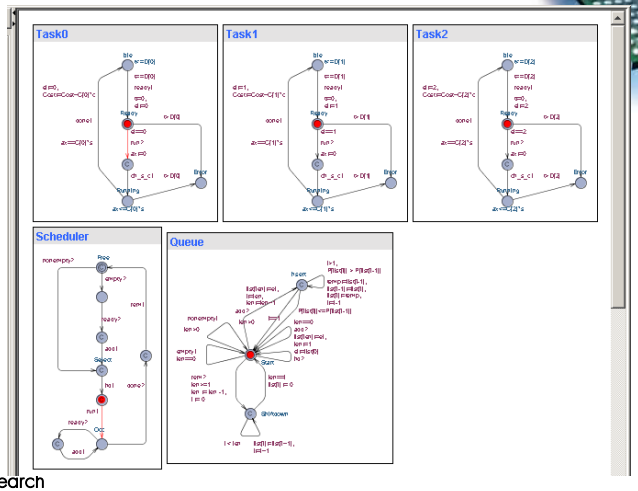
Application

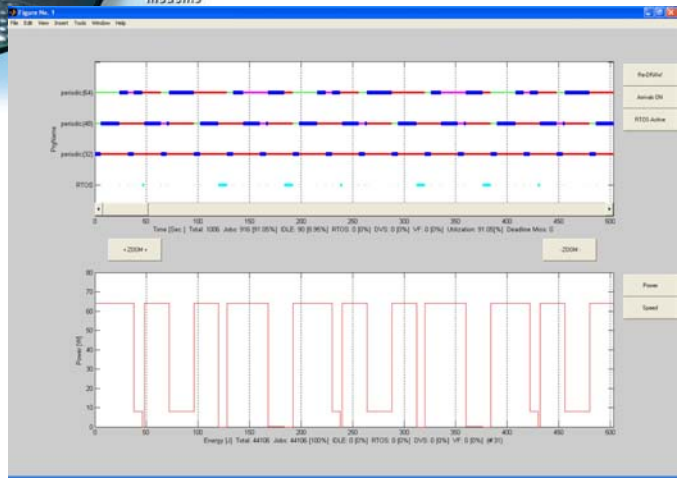
Dynamic Voltage Scaling

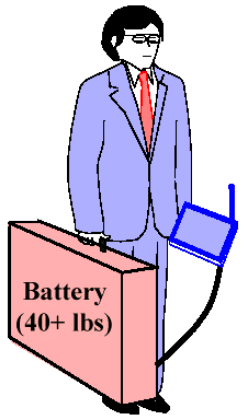






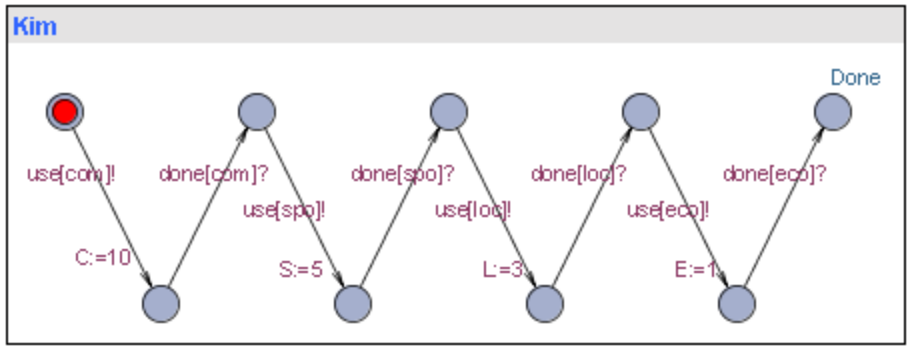






BRICS
Basic Research
in Computer Science

Dynamic Scheduling



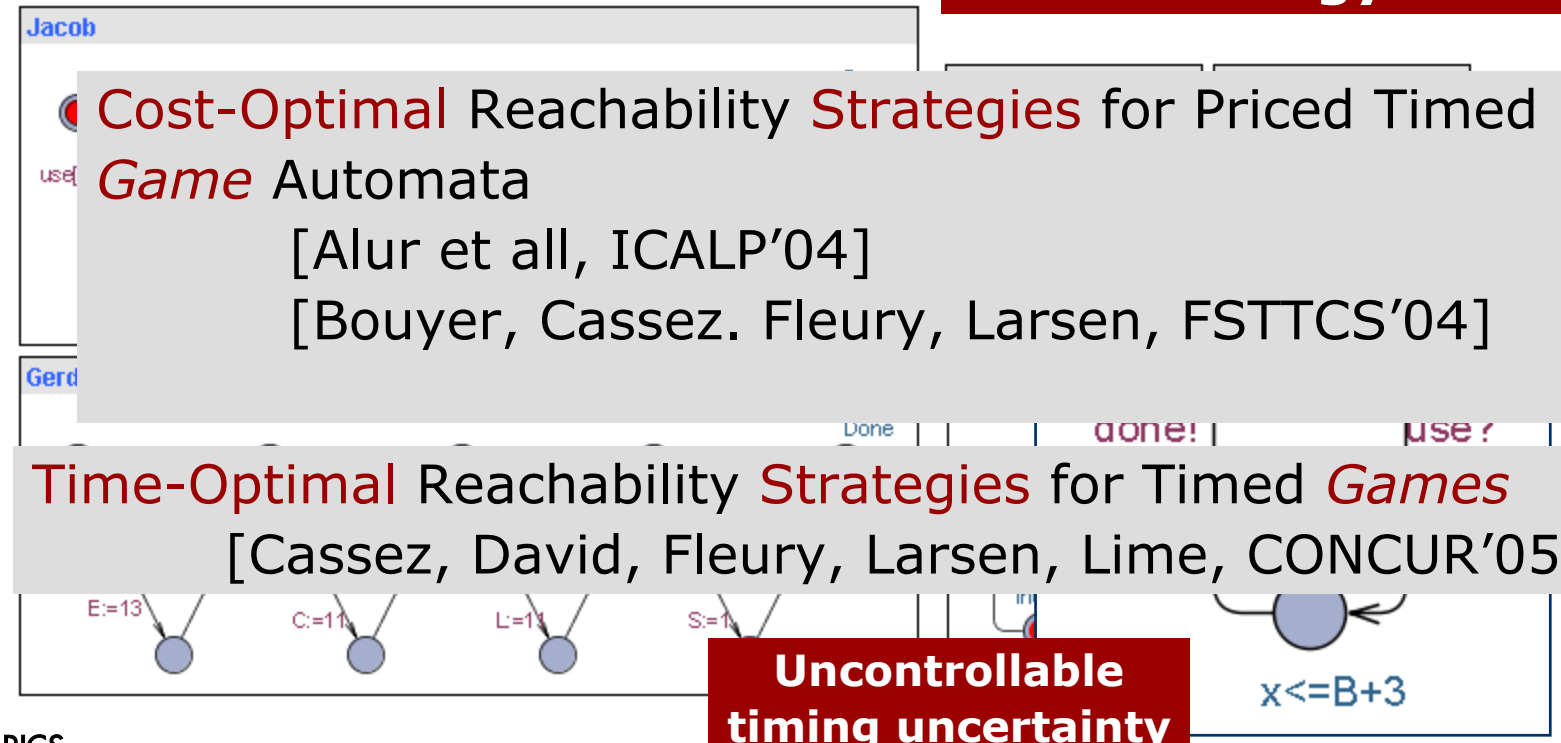
E<> (Kim.Done and
Jacob.Done and
Gerd.Done)

+ winning / optimal strategy

Cost-Optimal Reachability Strategies for Priced Timed *Game* Automata
[Alur et all, ICALP'04]
[Bouyer, Cassez. Fleury, Larsen, FSTTCS'04]

Time-Optimal Reachability Strategies for Timed *Games*
[Cassez, David, Fleury, Larsen, Lime, CONCUR'05]

Uncontrollable timing uncertainty



Future Work & Conclusion

- $PDDL_3$ versus PTA
- OPTOP versus UPPAAL Cora
- Planning versus Model Checking

- Improved heuristic search
  - FF's relaxed planning approach
  - pattern databases
  - beam search
  - externalization
  - ...

- Please visit www.uppaal.com

BRICS
Basic Research
in Computer Science

CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

Thank you for your attention !

☺ ☺ ☺

BRICS
Basic Research
in Computer Science